From Classical to Blockchain Consensus: What Are the Exact Algorithms?

Yanhong A. Liu
Stony Brook University
liu@cs.stonybrook.edu

Scott D. Stoller
Stony Brook University
stoller@cs.stonybrook.edu

ABSTRACT

This tutorial describes well-known algorithms for distributed consensus problems, from classical consensus to blockchain consensus, and discusses exact algorithms that are high-level as in pseudocode and directly executable at the same time. The tutorial consists of five parts:

- (1) An introduction to different distributed consensus problems, from classical consensus and Byzantine consensus to blockchain consensus.
- (2) An overview of well-known algorithms, from Paxos for classical consensus to the Bitcoin algorithm for blockchain consensus, including important variants such as Viewstamped Replication and Virtual Synchrony, as well as Proof-of-Stake vs. Proof-of-Work.
- (3) An overview of a method and language, DistAlgo, for expressing distributed algorithms precisely at a high-level as pseudocode and having them be directly executable at the same time.
- (4) A study of exact algorithms expressed at a high level for the most extensively studied algorithm variants, including Lamport's Paxos for classical consensus and Nakamoto's Bitcoin algorithm for blockchain consensus.
- (5) A demo of the direct execution of these exact algorithms by distributed processes.

CCS CONCEPTS

• **Computing methodologies** → **Distributed algorithms; Distributed programming languages**; • **Networks** → **Formal specifications**; • **Software and its engineering** → **Very high level languages; Concurrent programming structures; Specification languages**; • **Theory of computation** → *Distributed computing models*.

KEYWORDS

Distributed consensus, blockchain consensus, Paxos algorithm, Bitcoin algorithm, synchronization conditions, high-level queries

ACM Reference Format:

Yanhong A. Liu and Scott D. Stoller. 2019. From Classical to Blockchain Consensus: What Are the Exact Algorithms?. In *2019 ACM Symposium*

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(s).

PODC '19, July 29–August 2, 2019, Toronto, ON, Canada

© 2019 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-6217-7/19/07.

<https://doi.org/10.1145/3293611.3338022>

on *Principles of Distributed Computing (PODC '19)*, July 29–August 2, 2019, Toronto, ON, Canada. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3293611.3338022>

Distributed consensus

Distributed consensus is the problem of having a set of distributed processes, or agents, agree on a single value or a continuing sequence of values in the presence of faulty processes.

- Classical consensus considers failures where processes may crash and may later recover, and where messages may be lost, delayed, reordered, and duplicated.
- Byzantine consensus considers arbitrary failures, which may result from malicious actions, such as sending arbitrary messages.
- Blockchain consensus further considers Sybil attacks where a system is subverted by forging process identities.

Distributed consensus is at the core of distributed systems and services, which are increasingly indispensable in daily life.

- There is an increasing interest in algorithms for both classical consensus and blockchain consensus, especially the exact algorithms for deeper understanding and possible improvements by researchers, practitioners, and students.
- Especially desired are also languages and methods for expressing distributed algorithms and protocols precisely at a very high level, to facilitate the design and implementation, as well as analysis and verification, of these algorithms and protocols.

DistAlgo language

For expressing distributed algorithms at a high level, DistAlgo supports four main concepts:

- (1) distributed processes that can send messages,
- (2) high-level control flows for handling received messages, both synchronously and asynchronously,
- (3) high-level queries for synchronization conditions, and
- (4) declarative configuration for setting up and running.

DistAlgo minimally extends the Python programming language to support these concepts.

- DistAlgo has been used to express a wide variety of important distributed algorithms, including over a dozen well-known algorithms and variants for classical consensus and blockchain consensus.
- DistAlgo has been used by hundreds of students in graduate and undergraduate courses in dozens of different course projects, implementing the core of network protocols, distributed graph algorithms, distributed coordination services, distributed hash tables, distributed file systems, distributed

databases, parallel processing platforms, security protocols, and more.

- The algorithms and systems can be programmed much more easily and clearly compared to using conventional programming languages, e.g., in 30 lines instead of 300 lines, not to mention 3000 lines or more.

Tutorial materials

The tutorial materials are based on our studies of distributed algorithms, e.g., [1, 2], as well as many more algorithms studied in our ongoing research project, From Clarity to Efficiency for Distributed Algorithms, e.g., [3–5], after a dedicated workshop, Languages for Distributed Algorithms, organized by Black and Liu (<http://sites.google.com/site/ladameeting>)

- (1) Y. A. Liu, S. Chand, and S. D. Stoller. Moderately Complex Paxos Made Simple: High-Level Executable Specification of Distributed Consensus Algorithm. arXiv:1704.00082 [cs.DC], revised Jan. 2019. <https://arxiv.org/pdf/1704.00082v3.pdf>
- (2) Y. A. Liu, Logical Clocks Are Not Fair: What Is Fair? A Case Study of High-level Language and Optimization. In *Proceedings of the Workshop on Advanced Tools, Programming Languages, and Platforms for Implementing and Evaluating Algorithms for Distributed Systems (APPLIED)*, pages 21–27, Jul. 2018. ACM Press. <http://www.cs.stonybrook.edu/~liu/papers/LclockUnfair-APPLIED18.pdf>
- (3) Y. A. Liu, S. D. Stoller, B. Lin, and M. Gorbiovitski. From Clarity to Efficiency for Distributed Algorithms. In *Proceedings of 27th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA)*, pages 395–410, Oct. 2012. <http://www.cs.stonybrook.edu/~liu/papers/DistPL-OOPSLA12.pdf>
  - Extended description of DistAlgo language and method appears in *ACM Transactions on Programming Languages and Systems (TOPLAS)*, Volume 39, Number 3, May 2017. <https://arxiv.org/pdf/1412.8461v3.pdf>
  - Open-source implementation is available on github and sourceforge, version 1.0 released Nov. 2016. <http://sites.google.com/site/distalgo>
- (4) Y. A. Liu, S. D. Stoller, and B. Lin. High-Level Executable Specifications of Distributed Algorithms. In *Proceedings of the 14th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 7596 of LNCS, pages 95–110, Oct. 2012. <http://www.cs.stonybrook.edu/~liu/papers/DistSpec-SSS12.pdf>
  - This paper received the Best Student Paper Award.

Additional information and materials for the workshop will be made available at <http://darlab.cs.stonybrook.edu/consensus>, including presentation slides and executable specifications/programs in DistAlgo.

Biographies

Y. Annie Liu is a Professor of Computer Science at Stony Brook University. She received her BS from Peking University, MEng from Tsinghua University, and PhD from Cornell University. Her primary research is in languages and algorithms, and focuses specially on systematic methods for design and optimizations. The methods are centered around incrementalization, the discrete counterpart of differentiation in calculus. She has published in many prestigious venues, taught in a wide range of Computer Science areas, and presented over 100 conference and invited talks worldwide. Her

awards include a State University of New York Chancellor's Award for Excellence in Scholarship and Creative Activities.

Scott D. Stoller is a Professor of Computer Science at Stony Brook University. He received his BS in Physics, summa cum laude, from Princeton University and his PhD in Computer Science from Cornell University. His primary research interests are design, analysis, optimization, testing, and verification of software, with focuses on computer security, concurrency, and incremental computation. He received an NSF CAREER Award, an ONR Young Investigator Award, a NASA Turning Goals Into Reality Award for Engineering Innovation, Best Paper Awards at Haifa Verification 2005, Runtime Verification 2011, and Data and Applications Security and Privacy 2016, and a SUNY Chancellor's Award for Excellence in Scholarship and Creative Activities.

Acknowledgments

We thank Leslie Lamport, Robbert van Renesse, Yair Amir, and Ken Birman for their clear explanations and helpful discussions about various consensus algorithms. We thank Bo Lin for his robust DistAlgo compiler with excellent support. We thank hundreds of students in distributed algorithms and distributed systems courses and projects for extending, developing variants of, testing, evaluating, and model checking our executable specifications, including running them on distributed machines, in the cloud, etc.

This material is based upon work supported in part by the National Science Foundation under Grants CCF 1414078, CCF 1248184, and CNS 1421893, and Office of Naval Research under Grant N000141512208. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

- [1] Yanhong A. Liu, Saksham Chand, and Scott D. Stoller. 2019. Moderately Complex Paxos Made Simple: High-Level Executable Specification of Distributed Algorithm. *Computing Research Repository* arXiv:1704.00082 [cs.DC] (Jan. 2019), 24.
- [2] Yanhong A. Liu. 2018. Logical Clocks Are Not Fair: What Is Fair? A Case Study of High-level Language and Optimization. In *Proceedings of the Workshop on Advanced Tools, Programming Languages, and Platforms for Implementing and Evaluating Algorithms for Distributed Systems*. ACM Press, 21–27.
- [3] Yanhong A. Liu, Scott D. Stoller, Bo Lin, and Michael Gorbiovitski. 2012. From Clarity to Efficiency for Distributed Algorithms. In *Proceedings of the 27th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications*. ACM Press, 395–410.
- [4] Yanhong A. Liu, Scott D. Stoller, and Bo Lin. 2012. High-Level Executable Specifications of Distributed Algorithms. In *Proceedings of the 14th International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Springer, 95–110.
- [5] Yanhong A. Liu, Scott D. Stoller, and Bo Lin. 2017. From Clarity to Efficiency for Distributed Algorithms. *ACM Transactions on Programming Languages and Systems* 39, 3 (May 2017), 12:1–12:41.