Tutorial: Recent Progress in Practical State-Machine Replication Systems: Geo-Replication, Randomization, RDMA

LEWIS TSENG, Boston College, USA

State-Machine Replication (SMR) uses replication to ensure that a service is available and consistent in the presence of failures. Despite extensive research on SMR from both academia and industry for more than 30 years, there are still many unsolved challenges. This tutorial will first review the general concept of SMR, followed by the challenges in real-world systems. We will then focus on three exciting recent progresses in improving practical SMR systems: (i) adopting generalized consensus to shorten latency in geo-replicated systems; (ii) exploiting randomization to simplify the system design and engineering complexity without sacrificing performance; and (iii) using RDMA (Remote Direct Memory Access) to achieve micro-second latency in a single datacenter. Finally, we will discuss future research opportunities.

Additional Key Words and Phrases: SMR, Randomization, Geo-replication, RDMA

ACM Reference Format:
Lewis Tseng. 2018. Tutorial: Recent Progress in Practical State-Machine Replication Systems: Geo-Replication, Randomization, RDMA. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

State-Machine Replication (SMR) uses replication to ensure that a service is available and consistent in the presence of failures. One popular mechanism for implementing SMR is to use a consensus algorithm to agree on the total order of client requests (or commands), namely, a log-based SMR approach [5, 22, 24]. Paxos and variants [16, 17] had mostly been the de facto choice for implementing SMR, e.g., Chubby [7], Google Spanner [11], Microsoft Azure Storage [8], etc. Raft [22] recently becomes a popular alternative. Many modern production systems choose Raft over Paxos, e.g., Redis [2], RethinkDB [4], CockroachDB [3], and etcd [1]. Despite extensive research on SMR for more than 30 years, since Lamport's seminal work on Paxos [16, 17], building an SMR production systems still remains a tremendous challenge [7, 9, 15].

In this tutorial, we will share three exciting recent research directions in improving practical SMR systems: (i) adopting generalized consensus to shorten latency in geo-replicated systems; (ii) exploiting randomization to simplify the system design and engineering complexity without sacrificing performance; and (iii) using RDMA (Remote Direct Memory Access) to achieve micro-second latency in a single datacenter. Finally, we will discuss future research opportunities, especially potential applications in Blockchain and distributed ledgers.

2 SCOPE, MOTIVATION, SUMMARY OF TUTORIAL

Scope. This tutorial will focus on modern SMR systems, with focus on recent effort on bringing theoretical algorithms into real-world systems.

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

Motivation. SMR is one of the most used approaches to build a fault-tolerant distributed systems, and is the foundation of modern cloud infrastructures. Some of the concepts and tools could potentially be applied to other fault-tolerant distributed systems, specifically Blockchain and distributed ledgers.

Summary and Learning Goals. After the tutorial, the audiences are expected to learn:

- Brief introduction of the concept of log-based SMR and real-world SMR systems.
- Brief introduction of challenges in practical systems, ranging from theoretical challenges to engineering ones.
- Concept of generalized consensus, and recent geo-replicated SMR systems.
- Randomization tools developed for SMR systems, and recent systems using these tools to simplify design and engineering complexity.
- Brief introduction of RDMA and how it has been integrated with SMR systems to achieve micro-second latency.
- Unsolved challenges in practical SMR systems.

3 PAST TUTORIAL FROM THE SPEAKER

This will be a new tutorial, which will include our recent SOSP paper [23], our works under submission [21, 25], and many recent developments in SMR systems, e.g., [5, 6, 20]. The main speaker Lewis Tseng has a rich experience in tutorials and invited talks at premier conferences and workshops, including "Erasure Coding in Object Stores: Challenges and Opportunities" at PODC 2018, "Google Cloud for Beginners: Architecture, Storage, and Computation" and "Deep Dive into Data-Intensive Distributed Storage: Theory, Design, and Application" at PerCom 2020, and the keynote talk at FOCODILE 2021 ("Keynote: High-performance Crash-tolerant SMR in a Data Center").

4 TARGET AUDIENCE AND RELEVANCE TO SPAA 2022 RESEARCH TOPICS

Target Audience. The tutorial will be accessible to anyone with background of distributed systems and programming. We do *not* assume any background in fault-tolerant consensus, nor SMR systems. We welcome everyone to attend, but the tutorial is mainly designed for people who are interested in (i) understanding real-world challenges in SMR, (ii) developing high-performance fault-tolerant distributed systems, and (iii) understanding how recent progress on networking and hardware help improve real-world fault-tolerant systems.

Relevance. Fault-tolerant consensus is a classical problem in distributed computing. SMR is a key foundation of most modern cloud infrastructure. Hence, we believe the tutorial will be highly relevant for people who are generally interested in fault-tolerance issues. Moreover, all three relevant progresses touch both distributed computing and networking aspects (wide-area networks, stable network within a single datacenter, and remote memory); hence, this tutorial will hopefully attract attendances from both networking and distributed computing communities. Finally, the techniques presented in the paper can potentially be applied to "hotter areas," e.g., Blockchain and distributed ledgers.

5 TUTORIAL OUTLINE AND DURATION

The duration of the tutorial is 1.5 hours. The tutorial will follow the outline below:

- **Brief introduction of the concept of log-based SMR and real-world SMR systems.**
  - What is log-based SMR? Why is it important?
  - What is consensus? What are some important algorithms?

  - What are some real-world systems that are built on top of SMR?
- **Brief introduction of challenges in practical systems, ranging from theoretical challenges to engineering ones.**
  - Why is SMR/Consensus difficult?
  - Why is it challenging to deal with failure and asynchrony? (e.g., FLP impossibility [13])
  - Why is geo-replicated SMR important and difficult?
  - What are the bottlenecks of existing SMR systems?
  - Why is it difficult to implement, develop, and deploy an SMR system?
- **Concept of generalized consensus, and recent geo-replicated SMR systems.**
  - What is generalized consensus? Why is it relevant to geo-replication?
  - What are some breakthroughs in geo-replicated SMR systems? (e.g., EPaxos [20], Mencius [18], Atlas [12], Gryff [6], and Gus [25])
- **Randomization tools developed for SMR systems, and recent systems using these tools to simplify design and engineering.**
  - What is randomized consensus? Why is it appealing?
  - What are some recent systems that use randomization? (e.g., Algorand [10, 14] and HoneyBadger [19])
  - How does our work Rabia [23] use randomization to simply SMR?
- **Brief introduction of RDMA and how it has been integrated with SMR systems to achieve micro-second latency.**
  - What is RDMA? Why is it useful?
  - How can RDMA be used to solve consensus in micro-second latency?
  - How do our work [21] and Mu [5] develop Paxos-based SMR that achieves micro-second latency?
- **Unsolved challenges in practical SMR systems.**
  - What are the limitations of SMR systems mentioned earlier?
  - What are some possible solutions?
  - How can these techniques potentially be applied to Blockchain and distributed ledgers?

6 BIOGRAPHIES OF PRESENTERS

Lewis Tseng is an assistant professor in Computer Science department at Boston College. Before that, he spent a year and a half as a researcher at Toyota InfoTechnology Center. He received a B.S. and a Ph.D. degree both in Computer Science from the University of Illinois at Urbana-Champaign (UIUC) in 2010 and 2016, respectively. His research broadly lies in the intersection of fault-tolerant computing and distributed computing. He won the best paper award in the International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS) 2017. His group was the winner in the undergraduate category of Mobicom 2020 SRC (Student Research Competition) Award. He has been served in many committees including PODC '18, DISC '21, and ICDCS '21.

REFERENCES

[1] etcd: A distributed, reliable key-value store for the most critical data of a distributed system. https://etcd.io/, accessed May 2021.

[2] RedisRaft: Strongly-consistent Redis deployments. https://github.com/RedisLabs/redisraft, accessed December 2020.
[3] Replication Layer | CockroachDB Docs. https://www.cockroachlabs.com/docs/stable/architecture/replication-layer.html, accessed December 2020.
[4] RethinkDB 2.1: high availability. https://rethinkdb.com/blog/2.1-release/, accessed December 2020.
[5] Marcos K. Aguilera, Naama Ben-David, Rachid Guerraoui, Virendra J. Marathe, Athanasios Xygkis, and Igor Zablotchi. Microsecond consensus for microsecond applications. 2020.
[6] Matthew Burke, Audrey Cheng, and Wyatt Lloyd. Gryff: Unifying consensus and shared registers. In Ranjita Bhagwan and George Porter, editors, *17th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2020, Santa Clara, CA, USA, February 25-27, 2020*, pages 591–617. USENIX Association, 2020.
[7] Michael Burrows. The chubby lock service for loosely-coupled distributed systems. In Brian N. Bershad and Jeffrey C. Mogul, editors, *7th Symposium on Operating Systems Design and Implementation (OSDI '06), November 6-8, Seattle, WA, USA*, pages 335–350. USENIX Association, 2006.
[8] Brad Calder, Ju Wang, Aaron Ogus, Niranjan Nilakantan, Arild Skjolsvold, Sam McKelvie, Yikang Xu, Shashwat Srivastav, Jiesheng Wu, Huseyin Simitci, Jaidev Haridas, Chakravarthy Uddaraju, Hemal Khatri, Andrew Edwards, Vaman Bedekar, Shane Mainali, Rafay Abbasi, Arpit Agarwal, Mian Fahim ul Haq, Muhammad Ikram ul Haq, Deepali Bhardwaj, Sowmya Dayanand, Anitha Adusumilli, Marvin McNett, Sriram Sankaran, Kavitha Manivannan, and Leonidas Rigas. Windows azure storage: A highly available cloud storage service with strong consistency. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 143–157, New York, NY, USA, 2011. Association for Computing Machinery.
[9] Tushar Deepak Chandra, Robert Griesemer, and Joshua Redstone. Paxos made live: an engineering perspective. In Indranil Gupta and Roger Wattenhofer, editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing, PODC 2007, Portland, Oregon, USA, August 12-15, 2007*, pages 398–407. ACM, 2007.
[10] Jing Chen and Silvio Micali. Algorand: A secure and efficient distributed ledger. *Theor. Comput. Sci.*, 777:155–183, 2019.
[11] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson C. Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. Spanner: Google's globally-distributed database. In Chandu Thekkath and Amin Vahdat, editors, *10th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2012, Hollywood, CA, USA, October 8-10, 2012*, pages 251–264. USENIX Association, 2012.
[12] Vitor Enes, Carlos Baquero, Tuanir França Rezende, Alexey Gotsman, Matthieu Perrin, and Pierre Sutra. State-machine replication for planet-scale systems. In Angelos Bilas, Kostas Magoutis, Evangelos P. Markatos, Dejan Kostic, and Margo I. Seltzer, editors, *EuroSys '20: Fifteenth EuroSys Conference 2020, Heraklion, Greece, April 27-30, 2020*, pages 24:1–24:15. ACM, 2020.
[13] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.
[14] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*, pages 51–68. ACM, 2017.
[15] Tobias Grieger. Consensus, made thrive. https://www.cockroachlabs.com/blog/consensus-made-thrive/, 2016.
[16] Leslie Lamport. The part-time parliament. *ACM Transactions on Computer Systems (TOCS)*, 16(2):133–169, 1998.
[17] Leslie Lamport et al. Paxos made simple. *ACM Sigact News*, 32(4):18–25, 2001.
[18] Yanhua Mao, Flavio Paiva Junqueira, and Keith Marzullo. Mencius: Building efficient replicated state machine for wans. In Richard Draves and Robbert van Renesse, editors, *8th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2008, December 8-10, 2008, San Diego, California, USA, Proceedings*, pages 369–384. USENIX Association, 2008.
[19] Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The honey badger of BFT protocols. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 31–42. ACM, 2016.
[20] Iulian Moraru, David G. Andersen, and Michael Kaminsky. There is more consensus in egalitarian parliaments. In Michael Kaminsky and Mike Dahlin, editors, *ACM SIGOPS 24th Symposium on Operating Systems Principles, SOSP '13, Farmington, PA, USA, November 3-6, 2013*, pages 358–372. ACM, 2013.
[21] Jacob Nelson, Lewis Tseng, Roberto Palmieri, and Xiong Zheng. Boosting rdma models with atomic operations. In *Under Submission*, 2022.
[22] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm. In *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, USENIX ATC'14, page 305–320, USA, 2014. USENIX Association.

[23] Haochen Pan, Jesse Tuglu, Neo Zhou, Tianshu Wang, Yicheng Shen, Xiong Zheng, Joseph Tassarotti, Lewis Tseng, and Roberto Palmieri. Rabia: Simplifying state-machine replication through randomization. In *Proceedings of the 28th Symposium on Operating Systems Principles*, SOSP '21, 2021.

[24] Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Comput. Surv.*, 22(4):299–319, 1990.

[25] Lewis Tseng, Neo Zhou, Tigran Bantikyan, and Roberto Palmieri. Gus: Strongly consistent object storage across global data centers with optimal commit latency. In *Under Submission*, 2022.