Final Report: Intelligent Data Synchronization for Hybrid Clouds

David Starobinski
Professor of Electrical and Computer Engineering.
Systems Engineering, and Computer Science
Boston University
Boston MA 02215

Red Hat Collaboratory

January 2023

1 Results

Project objectives. Data synchronization is a core functionality for distributed systems to ensure (weak) consistency among different, possibly geographically distant components. This technology plays a key role in computing environments that must keep data synchronized across distributed and heterogeneous components. The current state-of-the-art in this field consists of a multitude of protocols leveraging various data structures and coding-theoretic constructions (IBLTs, cuckoo filters, characteristic polynomials, Bloom filters, etc.). The respective strengths and weaknesses of these methods are currently not well understood. This makes it hard to determine which among these methods is the best fit in any given network configuration.

To address the current gap, the PI and his collaborators have developed an open-source software package, dubbed `GenSync` [1], that unifies and enhances several state-of-the-art data synchronization protocols on a common programming platform. The GitHub repository of `GenSync` is available at [2].

The purpose of this project was to demonstrate the ability of `GenSync` to support configurable synchronization solutions on a common platform for a wide range of edge computing scenarios. A key goal was to deploy `GenSync` on the Colosseum testbed [3], located at Northeastern University, to demonstrate such benefits.

Main outcomes. This project resulted in several concrete outcomes:

- **Outcome 1: Survey of current data synchronization solutions.** We conducted a survey of the state-of-art of the literature and technical solutions for data synchronization in mobile and edge computing environments (e.g., IoT, fog computing, and 5G environments). We also explored how work from the literature made use of the Colosseum testbed. Our literature survey can be found in the Appendix.

- **Outcome 2: Implementation of `GenSync` into Colosseum.** We deployed `GenSync` on the Colosseum testbed. Since all user experiments in Colosseum are run in LXC containers, we installed the GenSync package within a container provided by the Colosseum team and updated it as needed to make adjustments.

  We used Colosseum's `Boston` cellular network scenario to emulate a cellular network in the vicinity of Boston Common in Boston, Massachusetts. The scenario has two regimes: (1) *Stationary:* user devices are stationary with respect to their base stations; and (2) *Pedestrian:* user devices are moving at 5 m/s relative to their base stations).

- **Outcome 3: Experimental results.** The resulting network traces (available bandwidth and latency) are plotted in Fig. 1, with their extremes annotated. These traces show that user movement results in wide oscillations of available bandwidth and latency. Using the average values of bandwidth and latency during the extreme periods, we define two sets of network conditions against which to evaluate our sync protocols: (1) *bad*: 1 Mb/s bandwidth and 50 ms latency; and (2) *good:* 7 Mb/s bandwidth and 30 ms latency. The resulting total sync times for the two sets of network conditions and three `genSync` protocols are plotted in Fig. 2.

  The main takeaway from our experiments is that the IBLT sync protocol [4, 5] performs the best in both good and bad network conditions. Hence, this protocol seems to be the best fit for typical edge computing environments.

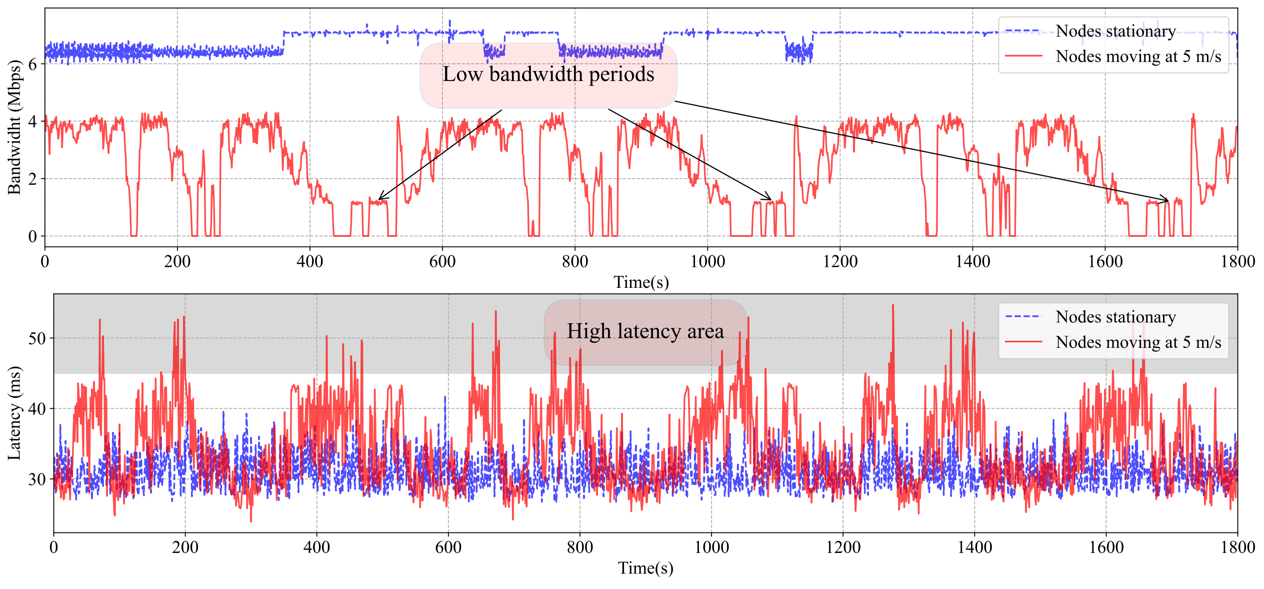

Figure 1: Bandwidth and latency traces for Colosseum's `Boston` scenario.

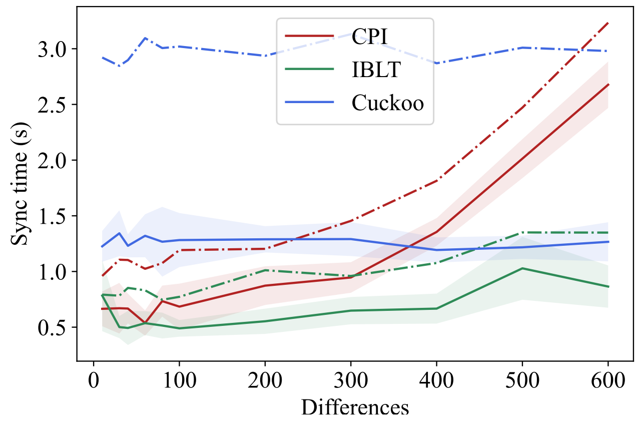

Figure 2: Sync time for three `GenSync` protocols (CPI, IBLT, and Cuckoo) under good network conditions (solid line) and under bad network conditions (dashed line).

- **Outcome 4: Paper writing and submission.** We have synthesized the results of this project into a paper that was submitted to a journal magazine [6]. This paper describes the APIs of `GenSync`, and includes the results of our Colosseum's experiments that showcase the benefits of `GenSync`.

2 Other highlights

- Şevval Şimşek attended the Young Gladiators workshop organized by the Colosseum team on June 6-8, 2022. Through this workshop, she learned the basics and use cases of the Colosseum testbed and had the opportunity to work along with the team. The workshop was very useful in helping Şevval gain a deeper insight into the working mechanism of the testbed.
- Our team met with the Colosseum team to introduce our project and discuss challenges that we faced while using the testbed, including unexpected failures of radio nodes and limitations

of the reservation system. The Colosseum team suggested workarounds to overcome some of these problems.

3 Participants

Funded participants under this project included:

- Prof. David Starobinski (PI).
- Şevval Şimşek (Graduate research assistant).

Collaborators included Prof. Ari Trachtenberg and Novak Boskov.

4 Input on Research Incubation Experience

1. We gratefully acknowledge the support of the Red Hat Collaboratory throughout the project.
2. The mid-term review was helpful in connecting us with the Colosseum research team.
3. We consider this project to be completed.
4. We are actively exploring other applications of `GenSync`, especially in the areas of blockchains and smart contracts. Prof. David Starobinski and Prof. Ari Trachtenberg recently received a $300K grant from NSF to conduct investigations in this area.

Appendix A Literature Survey

A.1 Data Synchronization at the Edge

The most primitive approach to data synchronization is executing a complete file transfer when a small change happens. Foreseeably, this method is extremely slow when working with large data sizes and causes a lot of redundant data transferred between the parties. To overcome this redundancy, Tridgell and Mackerras introduced the RSYNC [7] algorithm. The RSYNC algorithm includes a method to calculate the differences between two (or more) files by splitting the file into chunks and calculating the checksum values for these chunks. This way, first the checksums are sent to the other party in order to determine the modified chunks, and in the second step the modified chunks are sent over the network instead of the whole file.

Subsequently, Wang et al. [8] proposed an architecture for data synchronization based on fog computing, building on top of the RSYNC algorithm. They introduce fog computing as a middle computing layer between edge/IoT devices and cloud, which is similar to a cache server and called it Fog-Sync (FSYNC). By offloading part of the computing and storage work to the fog server, and with use of the differential synchronization algorithm, they reduce the communication cost and delay significantly. The FSYNC algorithm leverages the fog layer to minimize the number of synchronization requests using a threshold system, where the published changes are stored and distributed only when the threshold is reached. The authors further come up with an enhanced algorithm, which builds on top of the FSYNC algorithm and adds Reed-Solomon code to ensure security.

Ramsey and Csirmaz [9] analyze a scenario where multiple replicas of a file system are located on different devices. When a change is made on one of the replicas, the others do not have the current state of the document, so a file synchronizer makes them consistent again and preserves the changes. The authors propose an“algebra” of file system operations, designed to compute the sequence of operations needed in each replica to reach the common state, making the replicas consistent. This method is efficient, but a single round of communication creates a higher bandwidth need. Other works focus on lowering the bandwidth requirements by implementing multiple rounds, unlike RSYNC's single round synchronization method.

Yan et al. [10] claim that there are additional savings in bandwidth when multiple rounds are used and propose a new algorithm for file synchronization using set reconciliation methods for slow networks. For large collections and low bandwidth networks such as IoT and mobile devices, it is shown that multiple rounds further reduce communication costs, whereas a single round is more preferable for synchronizing small collections and in high bandwidth networks.

Suel et al. [11] propose a framework for synchronizing large, replicated collections over slow networks. Their method includes two phases, map construction and delta compression. During map construction, the two parties use a multi-round protocol to determine the common parts of the modified file(s). Then during delta compression, the remaining parts are encoded in relation to the common parts and transmitted to the other side. For efficient map construction, several techniques are used, such as recursive splitting, optimized match verification, local and continuation hashes, and decomposable hash functions. The prototype implementation achieves significant improvements over RSYNC, achieving 1.9 to 2.8 times faster reconciliation.

A.2 Colosseum Use Cases

The Colosseum testbed has been used in several projects focusing on Radio Access Networks (RANs). OrchestRAN [12] is an orchestration framework for next generation systems that builds upon the Open radio access network paradigm to serve a practical solution to these challenges. In this project, the authors use Colosseum testbed to test their prototype using NearRT RAN Intelligent Controller (RIC) containers and software-defined radios (SDRs). They use SCOPE [13], a softwarized prototyping platform, to generate datasets on Colosseum and train ML models.

The same team of authors create OpenRAN Gym [14], an open toolbox for data collection and experimentation with AI in Open Radio Access Network (O-RAN). This practical experimental toolbox provides end-to-end design, data collection, and testing workflows for intelligent control in next generation O-RAN systems. The authors demonstrate how the AI/ML solutions designed with OpenRAN Gym can be used to control a large-scale RAN simulated using SCOPE framework on the Colosseum wireless network emulator. The said RANs are managed by Open RAN near-RT intelligent controllers provided by the ColO-RAN [15] framework. The ColO-RAN framework is a large-scale O-RAN testing framework that builds on computational capabilities of the Colosseum wireless network emulator. This framework specifically leverages Colosseum as a wireless data factory to generate large-scale datasets for ML training in a variety of RF environments, benefiting from the realistic propagation and fading characteristics implemented in the Colosseum testbed.

ChARM [16] (Channel-Aware Reactive Mechanism) is a data-driven O-RAN compliant framework that senses the spectrum to infer the presence of interferences and reacts in real time. The authors use the Colosseum channel emulator to collect a large-scale waveform dataset to train their neural networks, and on top of that, to generate and collect Wi-Fi data, they extend the Colosseum testbed using system-on-chip (SoC) boards running a variant of OpenWiFi architecture. Similarly, another project by Camelo et. al [17] uses the Colosseum to generate a sample dataset of unknown radio technologies and measure their semi-supervised learning (SSL) algorithm accuracy.

QCell [18] is a model-free framework for cellular network self-optimization that determines the optimal scheduling policy for network slices and allocates resources. QCell is prototyped on Colosseum and trained and tested in a variety of network conditions and scenarios, using the base stations and radio nodes inside an urban scenario. The authors generate traffic for base stations and user equipment through iPerf3 to measure network performance and use Colosseum to recreate virtually indoor/outdoor environments and channel effects such as path loss, fading and mobility. Another Deep Q-Learning project [19] experimenting on Colosseum applies deep Q-Learning for dynamic spectrum sharing (DSS). They use one of Colosseum's ready scenarios, Alleys of Austin, mimicking mobile operations in an urban environment by five teams. The scenario is used to evaluate the deep Q-Learning algorithm to show that it can discover and take advantage of spatial distribution opportunities.

References

[1] N. Boškov, A. Trachtenberg, and D. Starobinski, “Gensync: A new framework for benchmarking and optimizing reconciliation of data,” *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[2] N. Boskov, A. Trachtenberg, D. Starobinski, and contributors. GenSync Framework. [Online]. Available: https://www.github.com/nislab/gensync

[3] Colosseum. [Online]. Available: https://www.colosseum.net

[4] M. T. Goodrich and M. Mitzenmacher, “Invertible bloom lookup tables,” in *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2011, pp. 792–799.

[5] D. Eppstein, M. T. Goodrich, F. Uyeda, and G. Varghese, “What’s the difference? efficient set reconciliation without prior context,” *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 218–229, 2011.

[6] N. Boškov, A. Trachtenberg, and D. Starobinski, “Synchronization: From files to data sets,” *IT Professional*, under revision.

[7] A. Tridgell and P. Mackerras, “The rsync algorithm,” 1996. [Online]. Available: https://openresearch-repository.anu.edu.au/bitstream/1885/40765/3/TR-CS-96-05.pdf

[8] T. Wang, J. Zhou, A. Liu, M. Z. A. Bhuiyan, G. Wang, and W. Jia, “Fog-based computing and storage offloading for data synchronization in iot,” *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4272–4282, 2019.

[9] N. Ramsey and E. Csirmaz, “An algebraic approach to file synchronization,” in *Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. ESEC/FSE-9. New York, NY, USA: Association for Computing Machinery, 2001, p. 175–185. [Online]. Available: https://doi.org/10.1145/503209.503233

[10] H. Yan, U. Irmak, and T. Suel, “Algorithms for low-latency remote file synchronization,” in *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, 2008, pp. 156–160.

[11] T. Suel, P. Noel, and D. Trendafilov, “Improved file synchronization techniques for maintaining large replicated collections over slow networks,” in *Proceedings. 20th International Conference on Data Engineering*, 2004, pp. 153–164.

[12] S. D’Oro, L. Bonati, M. Polese, and T. Melodia, “Orchestran: Network automation through orchestrated intelligence in the open RAN,” *CoRR*, vol. abs/2201.05632, 2022. [Online]. Available: https://arxiv.org/abs/2201.05632

[13] L. Bonati, S. D’Oro, S. Basagni, and T. Melodia, “Scope: An open and softwarized prototyping platform for nextg systems,” in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 415–426. [Online]. Available: https://doi.org/10.1145/3458864.3466863

[14] L. Bonati, M. Polese, S. D'Oro, S. Basagni, and T. Melodia, "Openran gym: An open toolbox for data collection and experimentation with ai in o-ran," 2022. [Online]. Available: https://arxiv.org/abs/2202.10318

[15] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Colo-ran: Developing machine learning-based xapps for open RAN closed-loop control on programmable experimental platforms," *CoRR*, vol. abs/2112.09559, 2021. [Online]. Available: https://arxiv.org/abs/2112.09559

[16] L. Baldesi, F. Restuccia, and T. Melodia, "Charm: Nextg spectrum sharing through data-driven real-time o-ran dynamic control," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 240–249.

[17] M. Camelo, A. Shahid, J. Fontaine, F. A. P. de Figueiredo, E. De Poorter, I. Moerman, and S. Latre, "A semi-supervised learning approach towards automatic wireless technology recognition," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019, pp. 1–10.

[18] B. Casasole, L. Bonati, S. D'Oro, S. Basagni, A. Capone, and T. Melodia, "Qcell: Self-optimization of softwarized 5g networks through deep q-learning," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021, pp. 01–06.

[19] J. M. Shea and T. F. Wong, "A deep q-learning dynamic spectrum sharing experiment," in *ICC 2021 - IEEE International Conference on Communications*, 2021, pp. 1–6.