Efficient (re)naming in Conflict-free Replicated Data Types (CRDTs)

Matthieu Nicolas
matthieu.nicolas@inria.fr

March 28, 2018

In order to serve an ever-growing number of users and provide an increasing volume of data, large scale systems such as data stores[3] or collaborative editing tools[4] have to adopt a distributed architecture. However, as stated by the CAP theorem[2], such systems cannot ensure both strong consistency and high availability in case of network partitions. As a result, literature and companies increasingly adopt the optimistic replication model known as eventual consistency[8] to replicate data among nodes. This consistency model allows replicas to temporarily diverge to be able to ensure high availability, even in case of network partition. Each node owns a copy of the data and can edit it, before propagating updates to others. A conflict resolution mechanism is however required to handle updates generated in parallel on different replicas.

An approach which gains in popularity since a few years proposes to define Conflict-free Replicated Data Types (CRDTs)[7]. These data structures behave as traditional ones, like *Set* or *Sequence* data structures, but are designed for a distributed usage. Their specification ensures that concurrent updates are resolved deterministically, without requiring any kind of agreement, and that replicas eventually converge immediately after observing some set of updates, thus achieving *Strong Eventual Consistency*[6].

Shapiro et al[7] present two designs of CRDTs : *State-based* CRDTs and *Operations-based* CRDTs.

State-based CRDTs define data structures whose states monotonically increase using idempotent and commutative merge functions. This allows one replica to share its local updates by broadcasting its state to others. Upon the reception of the state of another replica, a node is able to update its own state by merging them, regardless of its concurrent updates. Thanks to the properties of the defined state and merge function, states can be missed or delivered multiple times. As long as the most recent state of each replica is successfully broadcast to others once, each node will converge. Thus, no assumptions are made on the network layer. However, this is achieved by broadcasting the whole state repeatedly, which may be inefficient according to the size of the data structure.

Operations-based CRDTs define data structures with a set of operations to perform updates and a partial order between these operations, usually a causal order[5]. In addition, operations have to be designed such that concurrent one commute. This allows to propagate local updates by broadcasting corresponding operations to other replicas. Operations are delivered according to the defined partial order. Upon delivery of an operation, a replica updates its state by applying it. In comparison to *State-based* CRDTs, this solution achieves better performance, especially regarding the bandwidth consumption. Nevertheless, it requires the network layer to keep track of the defined partial order, which may be a complex and costly task.

To achieve convergence, *State-based* and *Operations-based* CRDTs proposed in the literature mostly rely on unique identifiers to reference updated elements. To generate such element identifiers, nodes often use their own identifiers as well as logical clocks. Thus, regarding to how node identifiers are generated, the size of element identifiers usually increases with the number of nodes. Furthermore, element identifiers have to comply to additional constraints according to the CRDT, for example forming a dense set in case of a sequence data structure[1]. In this case, element identifiers' size also increases according to the number of elements contained in the data structure. Therefore, the size of element identifiers is usually not bounded.

Since the size of identifiers is not bounded, the size of metadata attached to each element increases over time. It exceeds more and more the size of data itself. This impedes the adoption of CRDTs since nodes have to broadcast and store metadata, causing the application's performance and efficiency to decrease over time.

This PhD aims to address this issue. A first approach is to study identifiers proposed in the literature to list existing constraints on identifiers and their consequences on identifiers generation in order to propose more efficient specifications of identifiers. A second approach is to study this issue as a particular case of the renaming problem and to propose mechanisms to rename identifiers in order to reduce their size, still without requiring any kind of agreement between nodes.

References

[1] Luc André, Stéphane Martin, Gérald Oster, and Claudia-Lavinia Ignat. Supporting adaptable granularity of changes for massive-scale collaborative editing. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing - CollaborateCom 2013*, pages 50–59, Austin, TX, USA, October 2013. IEEE Computer Society. doi: 10.4108/icst.collaboratecom.2013.254123.

[2] Eric Brewer. Towards Robust Distributed Systems, 2000. URL https://people.eecs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf.

[3] The SyncFree Consortium. AntidoteDB: A planet-scale, available, transactional database with strong semantics. URL http://antidoteDB.eu/.

[4] Matthieu Nicolas, Victorien Elvinger, Gérald Oster, Claudia-Lavinia Ignat, and François Charoy. MUTE: A Peer-to-Peer Web-based Real-time Collaborative Editor. Proceedings of 15th European Conference on Computer-Supported Cooperative Work - Panels, Posters and Demos, pages 1–4, Sheffield, United Kingdom, August 2017. EUSSET. doi: 10.18420/ecscw2017_p5. URL https://hal.inria.fr/hal-01655438.

[5] Ravi Prakash, Michel Raynal, and Mukesh Singhal. An adaptive causal ordering algorithm suited to mobile computing environments. *Journal of Parallel and Distributed Computing*, 41(2):190–204, March 1997. ISSN 0743-7315. doi: 10.1006/jpdc.1996.1300. URL http://dx.doi.org/10.1006/jpdc.1996.1300.

[6] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free Replicated Data Types. In *International Symposium on Stabilization, Safety, and Security of Distributed Systems - SSS 2011*, pages 386–400, Grenoble, France, October 2011. Springer. doi: 10.1007/978-3-642-24550-3_29.

[7] Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. A comprehensive study of Convergent and Commutative Replicated Data Types. Research Report RR-7506, Inria – Centre Paris-Rocquencourt, January 2011. URL https://hal.inria.fr/inria-00555588.

[8] D. B. Terry, M. M. Theimer, Karin Petersen, A. J. Demers, M. J. Spreitzer, and C. H. Hauser. Managing update conflicts in bayou, a weakly connected replicated storage system. *SIGOPS Oper. Syst. Rev.*, 29 (5):172–182, December 1995. ISSN 0163-5980. doi: 10.1145/224057.224070. URL http://doi.acm.org/10.1145/224057.224070.

Matthieu NICOLAS

matthieu.nicolas@inria.fr

CAREER

PHD STUDENT | INRIA, COAST TEAM

October 2017 - Today | Nancy, France

EFFICIENT (RE)NAMING IN CONFLICT-FREE REPLICATED DATA TYPES

Conflict-free Replicated Data Types (CRDTs) are data structures behaving as traditional ones, like *Set* or *Sequence* data structures, but designed for a distributed usage. They are used in order to build large scale distributed systems adopting the optimistic replication model known as eventual consistency to replicate data among nodes. With this model, each node owning a copy of the data can edit it without any kind of coordination with other nodes. They then propagate the updates to others. The specification of CRDTs ensures that concurrent updates are resolved deterministically and that replicas eventually converge after observing all of them.
To achieve convergence, CRDTs proposed in the literature mostly rely on identifiers to reference updated elements. According to the kind of CRDT, identifiers have to comply to several constraints (unicity, forming a dense set...).
Because of these constraints, the identifiers size is often not bounded. Therefore, the size of metadata attached to each element increases with the number of updates. It thus exceeds more and more the size of data itself, decreasing the efficiency of the data structure over time.
The goal of this PhD is to address this issue by

- Proposing more efficient specifications of identifiers according to their set of constraints,
- Proposing mechanisms to rename identifiers to reduce their size.

RESEARCH & DEVELOPMENT SOFTWARE ENGINEER | INRIA, COAST TEAM

September 2014 – September 2017 | Nancy, France

PROJECT OPENPAAS::NG

The goal of this project is to design an open-source entreprise social network providing a suite of peer-to-peer collaborative office applications. The aim is to offer a reliable and free alternative to existing solutions such as Google Apps. This project is a joint work with the team DaSciM (Data Science and Mining) from the computer science laboratory from the Ecole Polytechnique, Linagora, XWiki SAS and Nexedi.
In this project, the COAST team works on topics such as the interorganisational federation of peer-to-peer systems and the securing of communications in this kind of collaboration. Furthermore, the team provides its expertise on eventually consistent data replication mechanisms in distributed systems.
In order to validate them, these works have been integrated in MUTE , peer-to-peer web based real-time collaboration editor.

- Maintaining of *LogootSplit*[3][4] implementation
- Study of the literature on Conflict-free Replicated Data Types and of their use cases.
- Development and integration of an anti-entropy mechanism[5]

Publications

[1] M. Nicolas, V. Elvinger, G. Oster, C.-L. Ignat, and F. Charoy. MUTE: A Peer-to-Peer Web-based Real-time Collaborative Editor. Proceedings of 15th European Conference on Computer-Supported Cooperative Work - Panels, Posters and Demos, pages 1–4, Sheffield, United Kingdom, Aug. 2017. EUSSET. doi: 10.18420/ecscw2017_p5. URL https://hal.inria.fr/hal-01655438.

[2] OpenPaaS::NG. EC1.1, Prototype : Infrastructure d'édition collaborative P2P. 2016.

[3] OpenPaaS::NG. SP2-L2.9, Rapport décrivant l'implantation du moteur temps réel v1. 2016.

[4] OpenPaaS::NG. SP2-L2.10, Rapport décrivant l'implantation du moteur temps réel v2. 2017.

[5] OpenPaaS::NG. SP2-L2.2, Rapport décrivant l'implantation du composant middleware de réplication hybride pair-à-pair v1. 2017.

ADT INRIA PLM

The PLM is an open-source programming exerciser. Developed by Gérald Oster and Martin Quinson, this application proposes to students to explore and to learn several concepts of the algorithmic through interactive and graphical exercises.
The goal of this project was to enhance this tool in an experimental platform dedicated to the teaching of computer science. To achieve this, a usage data collecting mechanism was required in order to generate a dataset. This dataset, made available to researchers, allows the realisation of research works on topics such as the design of an automatic helping tool which tailors error messages to students. A second objective of this project was to port the application, until then released as a desktop Java software, into a web application to make it available to the greatest number.
My works focused mainly on the realisation of that port. This major change of paradigm introduced several issues which needed to be addressed.

- Implementation and testing of the usage data collecting mechanism
- Conception and integration of a distributed architecture ensuring the scalability of the application
- Isolation of the execution of student code using containers
- Deployment and monitoring of a multi-components application

INTERN | UNIVERSITY OF LORRAINE, COAST TEAM

April 2014 – August 2014 | Nancy, France

Design of a web based real-time collaborative editing tool

Coming from the research on collaborative editing, a new family of data replication and consistency maintenance algorithms was recently formalised : Conflict-free Replicated Data Types (CRDTs) approach. This new family addresses several unresolved issues from other approaches, especially its scalability.
The COAST team proposed a new algorithm from this family : *LogootSplit*.
In order to illustrate and highlight the work of the team on this new approach, my task was to design and implement a new real-time collaborative editing tool based on this algorithm.

- Implementation of *LogootSplit* as a library
- Design and development of MUTE , a real-time collaborative editing web app using this library

INTERN | POLYTECHNIQUE MONTRÉAL

September 2009 – June 2011 | Montreal, Canada

Development of a tool to check the correctness of collaborative editing algorithms

Existing collaborative editing tools relies mostly on a specific family of algorithms to ensure the eventual consistency of copies : the operational transformation.
Two consistency properties *TP1* and *TP2* are defined and allow to ensure the correctness of algorithms from this family.
The goal of this internship was to develop a tool to automatically check the respect of these properties for a given algorithm.

- Implementation of several algorithms from the operational transformation family
- Development of the tool allowing to check if the algorithms ensure *TP1* and *TP2*

EDUCATION

MASTER DEGREE IN COMPUTER SCIENCE

Engineering degree in Computer Science at TELECOM Nancy
September 2011 – August 2014 | Nancy, France

COMMUNICATION

I presented our research works on various occasions using MUTE. The aims of these presentations were to describe the issues encountered in nowadays systems (scalability, availability, privacy) and how CRDTs allow us to address them. These demos were also the opportunity to show the new usages introduced by this technology such as the seamless transition between real-time and asynchronous collaboration.

August 2017	ECSCW 2017 : MUTE : A Peer-to-Peer Web-based Real-time Collaborative Editor.
December 2016	HCERES Evaluation of the LORIA
	Inria Industry Meeting "New technologies to protect digital data and computer systems"
	Visit of a delegation of Technological University presidents from Mexico
November 2016	Inria Industry Meeting "Interaction with digital objects and services"
October 2016	Evaluation seminar of Inria teams working on "Distributed Systems and Middleware"