SyncFree

SyncFree Technology White Paper

Verifico: CRDT-App Verification Framework for Isabelle

Peter Zeller
U. Kaiserslautern

22 December 2016

The need for verification tools

Programming applications using weak consistency is inherently complex. Most importantly, convergence must be ensured, meaning that all replicas represent the same abstract state when they have observed the same set of operations, without losing writes. To help programmers handle this problem, conflict-free replicated data types (CRDTs) have been developed. A CRDT is a reusable data type, which embodies a certain strategy to handle concurrent updates. Examples are counters, sets, and maps. When an application is written using CRDTs, the convergence property comes for free and thus the development effort is reduced.

However, convergence is not the only desirable property of an application. It is also important that concurrent updates are handled in a way that makes sense for the application. These correctness properties are often overlooked by developers. One reason for this is that there is no systematic method to reason about the correctness of an implementation. While there are multiple program logics for working with sequential and concurrent programs, there are no frameworks yet, which support reasoning about eventual consistency and CRDTs on a higher level. Thus, it is not feasible to use existing frameworks to reason about non-trivial correctness properties of these kinds of applications.

Our approach is currently targeted at developers experienced in formal verification. Our goal is to start with a general approach, which can be used to verify any functional property for a very wide range of applications. This distinguishes our framework from automated approaches, which can be used for a small set of properties (e.g. integrity constraints on the data) and for a more limited set of applications (e.g. no loops and exactly one transaction per operation). Automation can be added via extensions to the basic framework, which we plan to provide with future releases.

The framework can also be used to prove general programming patterns correct, which then can be used by other programmers, which do not have a background in formal verification.

Main contributions:

- A formal semantics of a replicated database.
- Composable CRDT specifications.
- Infrastructure for proving applications correct (proof rules, useful simplifications, etc.)

Relation to other Syncfree products:

- The framework includes a formal semantics of a replicated database with parallel snapshot transactions, causal consistency, and CRDTs. These are exactly the guarantees provided by Antidote, so the framework can be used to reason about the correctness of an application that uses Antidote.
- CISE is a related tool developed in the SyncFree project, which provides more automation, but is less flexible.

Usage

The framework is developed in Isabelle/HOL.[1] To verify an application, it has to be modeled in a simple language, which is embedded into Isabelle and provided by our framework. Then the desired properties can be specified using logical formulas.

In general, it is not decidable whether a given application satisfies a property, so the verification developer has to explain why the application satisfies the desired property. This explanation is done by specifying additional invariants.

The framework then uses this input to derive verification conditions. Those can be proven interactively using the Isabelle proof system. The framework provides simplification rules to support this task.

More information is available at:
https://softech-git.cs.uni-kl.de/zeller/isabelle_crdt_apps.

We also refer the reader to the companion white paper: " Just-Right Consistency, or How to tailor consistency to application requirements."

[1] https://isabelle.in.tum.de