Formal Verification of Consensus Algorithms in a Proof Assistant

Henri Debrat, Bernadette Charron-Bost, Stephan Merz

Formal Verification of Consensus Algorithms in a Proof Assistant

Henri Debrat
Nancy University – LORIA
Nancy, France
Henri.Debrat@loria.fr

Bernadette Charron-Bost
CNRS – LIX
Palaiseau, France
charron@lix.polytechnique.fr

Stephan Merz
INRIA – LORIA
Nancy, France
Stephan.Merz@loria.fr

I. Introduction

Distributed algorithms are often quite subtle, both in the way they operate and in the assumptions under which they work correctly. Indeed, several algorithms have been found to be erroneous, and numerous misunderstandings have arisen due to different interpretations of the precise objectives of the algorithms and of the underlying hypotheses. Formal verification is therefore crucial in distributed computing.

In a distributed system, individual processing *nodes* communicate among each other by exchanging messages along point-to-point channels or *links*. Standard interleaving models represent executions of distributed algorithms as linear sequences of events local to the nodes: local computation, message sending and reception. It is well known that this fine-grained semantics leads to state explosion in model checking approaches to verification, making it impractical to verify more than small finite instances. It also complicates deductive verification, because all possible intermediate system states must be accounted for in the invariant that is at the base of correctness proofs, yielding large and hard to understand invariants.

We are therefore interested in models that allow us to verify algorithms over a coarser-grained representation, taking advantage of the *causal order* [1] between events of a distributed system instead of explicitly considering all possible linearizations. An important insight comes from the work of Elrad and Francez [2] who proposed to structure distributed algorithms into *rounds* that constitute *communication-closed layers*: messages are delivered only in the round in which they were sent. Indeed, many distributed algorithms are based on this principle, which helps algorithm designers to understand the possible global configurations a system can be in.

Consensus is regarded as the fundamental problem that must be solved to implement a fault-tolerant distributed system. It requires nodes to eventually agree on a common value among the initial values held by each of them. Agreement should be reached despite the failures of some components (process or link). Depending on the timing and failure models, numerous Consensus algorithms have been proposed in the literature. We are studying techniques for proving the correctness of these algorithms using the interactive proof assistant Isabelle [3], aiming for a high level of automation.

II. Benign errors: the HO model

Charron-Bost and Schiper [4] proposed the Heard-Of (HO) model of distributed algorithms that tolerate failures. In marked contrast to standard models, the HO model does not record the failures that occur during an execution, but focuses on what "goes well". The authors show that in this way the model can uniformly represent all *benign* errors, which include process crashes and message losses.

More precisely, an algorithm in the HO model is represented as an infinite sequence of communication-closed rounds. In each round, every node sends messages to all other nodes, receives messages from other nodes sent in the current round, and processes them in a local computation step. The set of nodes from which node n receives messages in round r is called the *heard-of* set $HO(n, r)$. Assumptions about the failures tolerated by the algorithm are captured by a *communication predicate*, which is expressed in terms of the sets $HO(n, r)$ that occur during an execution, and correctness is formally asserted relative to specific communication predicates.

The HO model thus combines the communication-closed round structure, which underlies most (if not all) known Consensus algorithms, with a uniform and elegant way of representing failure models in terms of correctness hypotheses and properties. Important properties of algorithms, including Consensus, can be verified in terms of a coarse execution model that models entire rounds as being performed atomically [5]. This allows us to construct finite-state representations of executions for finite algorithm instances, despite the presence of unbounded round numbers, and to validate algorithms using model checking.

We have represented [6] the HO model as a *locale* in the interactive proof assistant Isabelle/HOL, and have since instantiated this model for several Consensus algorithms that appear in [4]. Making use of the coarse-grained representation of HO algorithms, and of generic lemmas proved once and for all in the Isabelle locale, the verification of a new algorithm now takes about one person-week, and our proofs are shorter by roughly an order of magnitude compared to

proofs of similar algorithms in an interleaving model [7]. Most importantly, we find the proofs much more readable, and indeed comparable to a careful pencil-and-paper proof.

III. Towards malicious errors: the SHO model

The HO model does not account for so-called *value faults* (sometimes called malicious or Byzantine faults) that result from values being corrupted during local computation or message transmission. In this case, a node n receiving a message m from some node n' has no guarantee that m corresponds to what n' should have sent according to the algorithm, be it that n' is deliberately cheating, that a fault occurred during its computation, or that the message was corrupted during transmission.

Biely et al. [8] proposed an extension of the HO model that takes into account these more general kinds of faults. The basic idea is again to not identify a "culprit" responsible for the fault, and to limit failures to transmission faults. Indeed, nodes are supposed to perform state transitions correctly, i.e., following their program code. In order to express the discrepancies between the values that should have been sent and those that have been received, the model introduces sets $SHO(n, r)$ of all nodes n' from which n receives the expected message for round r. Assumptions on the occurrence of benign and value faults are formally stated in terms of the sets $HO(n, r)$ and $SHO(n, r)$. Because a node n cannot determine if a value fault has occurred, it cannot refer to the sets $SHO(n, r)$ in its code.

We are currently validating this model by encoding in it some standard Consensus algorithms that tolerate value faults. We will then extend the existing Isabelle locale to take into account the SHO model. In future work, we are mainly interested in two aspects: (1) a more detailed study of the properties that are preserved by the reduction to coarse-grained representations of runs, and (2) a higher degree of automation in verifying such algorithms, for example by representing elementary data structures such as HO sets in theories that are amenable to automatic reasoning using SMT solvers or saturation-based theorem provers.

References

[1] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, Jul. 1978.

[2] T. Elrad and N. Francez, "Decomposition of distributed programs into communication-closed layers," *Science of Computer Programming*, vol. 2, no. 3, pp. 155–173, 1982.

[3] T. Nipkow, L. Paulson, and M. Wenzel, *Isabelle/HOL. A Proof Assistant for Higher-Order Logic*, ser. Lecture Notes in Computer Science. Springer Verlag, 2002, no. 2283.

[4] B. Charron-Bost and A. Schiper, "The Heard-Of model: computing in distributed systems with benign faults," *Distributed Computing*, vol. 22, no. 1, pp. 49–71, 2009.

[5] M. Chaouch-Saad, B. Charron-Bost, and S. Merz, "A reduction theorem for the verification of round-based distributed algorithms," in *Reachability Problems '09*, ser. Lecture Notes in Computer Science, O. Bournez and I. Potapov, Eds., vol. 5797. Palaiseau, France: Springer, 2009, pp. 93–106.

[6] B. Charron-Bost and S. Merz, "Formal verification of a Consensus algorithm in the Heard-Of model," *Intl. J. Software and Informatics*, vol. 3, no. 2-3, pp. 273–204, 2009.

[7] M. Jaskelioff and S. Merz, "Proving the correctness of Disk Paxos," Archive of Formal Proofs, http://afp.sourceforge.net/entries/DiskPaxos.shtml, 2005.

[8] M. Biely, J. Widder, B. Charron-Bost, A. Gaillard, M. Hutle, and A. Schiper, "Tolerating corrupted communication," in *Proc. 26th ACM Symp. Principles of Distributed Computing (PODC 2007)*. Portland, Oregon, USA: ACM, 2007, pp. 244–253.