Applying the EPR verification methodology to Casper

Giuliano Losa

January 24, 2025

Contents

This is an experiment dating from 2017 in which I shortened an Isabelle/HOL proof by Yoichi Hirai from about 1000 lines to less than 100 by encoding things in EPR as much as possible.

It seems that this proof was later re-used by Yoichi Hirai and then translated to Coq by Runtime Verification:

https://github.com/runtimeverification/casper-proofs/blob/master/Core/AccountableSafety.v.

theory *Casper*
  imports *Main*
begin

Here we prove that the slashing conditions in Casper, as described at the url below by Vitalik Buterin, are such that if there is a fork, then one third of the validators are "slashed". https://medium.com/@VitalikButerin/minimal-slashing-conditions-20f0b50

We use first-order modeling as much as possible. This allows to reduce the size of the model, and also the size of the proofs from more than 1000 lines in Yoichi's proof to less than a 100.

locale *byz-quorums* =
  — "Here we fix two types $'q1$ and $'q2$ for quorums of cardinality greater than 2/3 of the validators and quorum of cardinality greater than 1/3 of the validators.
  fixes *member-1* :: $'n \Rightarrow 'q1 \Rightarrow bool$ (**infix** $\in_1$ *50*)
    — Membership in 2/3 set
    and *member-2* :: $'n \Rightarrow 'q2 \Rightarrow bool$ (**infix** $\in_2$ *50*)
    — Membership in 1/3 set
  assumes $\bigwedge q1\ q2 .\ \exists\ q3 .\ \forall\ n .\ n \in_2 q3 \longrightarrow n \in_1 q1 \wedge n \in_1 q2$
    — This is the only property of types $'q1$ and $'q2$ that we need: 2/3 quorums have 1/3 intersection

record $('n,'h)state$ =
  — "$'n$ is the type of validators (nodes), $'h$ hashes, and views are *nat*
  commit-msg :: $'n \Rightarrow 'h \Rightarrow nat \Rightarrow bool$
  prepare-msg :: $'n \Rightarrow 'h \Rightarrow nat \Rightarrow nat \Rightarrow bool$

locale *casper* = *byz-quorums* +
  — Here we make assumptions about hashes. In reality any message containing a hash not satisfying those should be dropped.
fixes
  hash-parent :: $'h \Rightarrow 'h \Rightarrow bool$ (**infix** $\leftarrow$ *50*)
assumes
  — "a hash has at most one parent which is not itself
  $\bigwedge h1\ h2 .\ h1 \leftarrow h2 \Longrightarrow h1 \neq h2$
  and $\bigwedge h1\ h2\ h3 .\ [\![h2 \leftarrow h1;\ h3 \leftarrow h1]\!] \Longrightarrow h2 = h3$
begin

lemmas *casper-assms-def* = *casper-def casper-axioms-def byz-quorums-def*

inductive *hash-ancestor* (**infix** $\leftarrow^*$ *50*) **where**
  $h1 \leftarrow h2 \Longrightarrow h1 \leftarrow^* h2$
| $[\![h1 \leftarrow h2;\ h2 \leftarrow^* h3]\!] \Longrightarrow h1 \leftarrow^* h3$
declare *hash-ancestor.intros*[*simp*,*intro*]
lemma *hash-ancestor-intro'*: **assumes** $h1 \leftarrow^* h2$ **and** $h2 \leftarrow h3$ **shows** $h1 \leftarrow^* h3$
  ⟨*proof*⟩

inductive *nth-ancestor* :: $nat \Rightarrow 'h \Rightarrow 'h \Rightarrow bool$ **where**
  nth-ancestor 0 h1 h1

| $[\![$*nth-ancestor n h1 h2*; *h2* $\leftarrow$ *h3*$]\!] \Longrightarrow$ *nth-ancestor* (*n*+*1*) *h1 h3*
declare *nth-ancestor.intros*[*simp*,*intro*]
inductive-cases *nth-ancestor-succ*:*nth-ancestor* (*n*+*1*) *h1 h3*
inductive-cases *zeroth-ancestor*:*nth-ancestor 0 h1 h3*
lemma *parent-ancestor*:*h1* $\leftarrow$ *h2* = *nth-ancestor 1 h1 h2*
$\langle$*proof*$\rangle$

All messages in epoch *0* are ignored; *0* is used as a special value (was − *1* in the original model).

definition *prepared′* **where**
prepared′ s q h v1 v2 $\equiv$ *v1* $\neq$ *0* $\wedge$ ($\forall$ *n* . *n* $\in_1$ *q* $\longrightarrow$ *prepare-msg s n h v1 v2*)
definition *prepared* **where**
prepared s q h v1 v2 $\equiv$ *v1* $\neq$ *0* $\wedge$ *v2* < *v1* $\wedge$ ($\forall$ *n* . *n* $\in_1$ *q* $\longrightarrow$ *prepare-msg s n h v1 v2*)
definition *committed* **where**
committed s q h v $\equiv$ *v* $\neq$ *0* $\wedge$ ($\forall$ *n* . *n* $\in_1$ *q* $\longrightarrow$ *commit-msg s n h v*)
definition *fork* **where**
fork s $\equiv$ $\exists$ *h1 h2 q1 q2 v1 v2* . *committed s q1 h1 v1* $\wedge$ *committed s q2 h2 v2*
$\wedge$ $\neg$(*h2* $\leftarrow^*$ *h1* $\vee$ *h1* $\leftarrow^*$ *h2* $\vee$ *h1* = *h2*)

definition *slashed-1* **where** *slashed-1 s n* $\equiv$
$\exists$ *h v* . *commit-msg s n h v* $\wedge$ ($\forall$ *q v2* . *v2* < *v* $\longrightarrow$ $\neg$*prepared s q h v v2*)
definition *slashed-2* **where**
slashed-2 s n $\equiv$
$\exists$ *h v1 v2* . *prepare-msg s n h v1 v2* $\wedge$ *v2* $\neq$ *0* $\wedge$
($\forall$ *v3 q h2* . *v3* < *v2* $\longrightarrow$ $\neg$(*nth-ancestor* (*v1* − *v2*) *h2 h* $\wedge$ *prepared s q h2 v2 v3*))
definition *slashed-3* **where**
slashed-3 s n $\equiv$
$\exists$ *h1 h2 v1 v2 v3* . *v1* < *v2* $\wedge$ *v3* < *v1* $\wedge$
commit-msg s n h1 v1 $\wedge$ *prepare-msg s n h2 v2 v3*
definition *slashed-4* **where**
slashed-4 s n $\equiv$ $\exists$ *h1 h2 v v1 v2* . (*h1* $\neq$ *h2* $\vee$ *v1* $\neq$ *v2*) $\wedge$
prepare-msg s n h1 v v1 $\wedge$ *prepare-msg s n h2 v v2*
definition *slashed* **where** *slashed s n* $\equiv$
slashed-1 s n $\vee$ *slashed-2 s n* $\vee$ *slashed-3 s n* $\vee$ *slashed-4 s n*
definition *one-third-slashed* **where** *one-third-slashed s* $\equiv$ $\exists$ *q* . $\forall$ *n* . *n* $\in_2$ *q* $\longrightarrow$ *slashed s n*

lemmas *slashed-defs* = *slashed-def slashed-1-def slashed-2-def slashed-4-def slashed-3-def one-third-slashed-def*
lemmas *order-defs* = *class.linorder-axioms-def class.linorder-def class.order-def class.preorder-def*
class.order-axioms-def class.order-bot-def class.order-bot-axioms-def linorder-axioms[**where** *?′a*=*nat*]
lemmas *casper-defs* = *slashed-defs prepared-def fork-def committed-def casper-assms-def*

lemma *l1*: **assumes** *prepared s q1 h1 v1 v2* **and** *committed s q2 h2 v3* **and** *v1* > *v3* **and** $\neg$*one-third-slashed s*

shows *v1* > *v2* ∧ *v2* ≥ *v3* ⟨*proof*⟩

lemma *l2*: **assumes** *nth-ancestor n h1 h2* **and** *nth-ancestor m h2 h3*
shows *nth-ancestor* (*n*+*m*) *h1 h3*
⟨*proof*⟩

lemma *l3*: **assumes** *prepared s q1 h1 v1 v2* **and** *committed s q2 h2 v3* **and** *v1* > *v3* **and** ¬*one-third-slashed s*
shows *nth-ancestor* (*v1* − *v3*) *h2 h1*
⟨*proof*⟩

lemma *l4*:**assumes** *nth-ancestor n h1 h2* **shows** *h1* ←* *h2* ∨ *h1* = *h2* ⟨*proof*⟩

lemma *safety*: **assumes** *fork s* **shows** *one-third-slashed s*
⟨*proof*⟩

end

end