Dynamically-available consensus with 1/2 failures in Isabelle/HOL

Giuliano Losa

August 31, 2023

Contents

theory *DynamicConsensus*
  imports *Main*
begin

1 Lemmas about majorities

definition *majority* **where** — A is a strict majority among B
  $majority\ A\ B \equiv A \subseteq B \wedge 2*(card\ A) > card\ B$

To make proofs simpler we assume that the set of processors is finite. We could rewrite the proofs using explicit finiteness assumptions when needed (e.g. the set of participants is always finite)

lemma *majorities-intersect*:
  fixes $A\ B\ C$ :: $('p::finite)\ set$
  assumes $C \neq \{\}$ **and** $majority\ A\ C$ **and** $majority\ B\ C$
  shows $A \cap B \neq \{\}$
proof (*rule ccontr*; *simp*) — proof by contradiction
  assume $A \cap B = \{\}$
  hence $card\ C \geq card\ A + card\ B$
    by (*metis Un-least assms*(*2*) *assms*(*3*) *card-Un-disjoint card-mono finite-code majority-def*)
  moreover
  have $2*(card\ A + card\ B) > 2*(card\ C)$

by (*metis add-less-mono assms*(*2*) *assms*(*3*) *distrib-left-numeral majority-def mult-2*)
ultimately
show *False* **by** *auto*
qed

lemma *majority-anti*:
fixes $A\ B\ C$ :: ($'p$::*finite*) *set*
assumes $C \neq \{\}$ **and** *majority* $A\ C$ **and** $B \cap A = \{\}$
shows *majority* A $(C-B)$
by (*smt* (*verit*, *ccfv-SIG*) *Diff-eq Diff-subset Int-greatest assms*(*2*) *assms*(*3*) *card-mono compl-le-swap1 finite-code inf-shunt majority-def order-le-less-trans*)

lemma *maj-increasing*:
assumes (A::$'p$::*finite set*) $\subseteq B$ **and** $B \subseteq X$ **and** $\neg$*majority* $B\ X$
shows $\neg$*majority* $A\ X$
proof $-$
have *card* $A \leq$ *card* B
by (*simp add*: *assms*(*1*) *card-mono*)
thus *?thesis*
using *assms* **unfolding** *majority-def* **by** *auto*
qed

lemma *card-maj-gt-card-not-maj*:
assumes *majority* $A\ X$ **and** $B \subseteq X$ **and** $\neg$ *majority* $B\ X$
shows *card* $A >$ *card* B
by (*smt* (*verit*, *ccfv-threshold*) *assms*(*1*) *assms*(*2*) *assms*(*3*) *linorder-neqE-nat majority-def nat-mult-less-cancel-disj order-less-subst1*)

2 Definition of a round of the No-Equivocation Model

locale *pre-round* =
fixes
P — The participating set for the round
F — The faulty set
C :: ($'p$::*finite*) *set* — The set of participating, non-faulty processors
and
HO :: $'p \Rightarrow 'p$ *set* — Maps each processor to the set of processors it hears of
and
bcast :: $'p \Rightarrow 'm$ — *bcast* $p = m$ means p broadcasts m.
and
rcvd :: $'p \Rightarrow 'p \Rightarrow 'm$ — *rcvd* $p\ q = m$ means p receives m from q
and
lambda :: $'m$ — Failure indication
defines $C \equiv P-F$
begin

notation *lambda* (λ)

end

locale *round* = *pre-round* +
  assumes
    p2:*majority C P* — majority correct
    and *p3*:$\bigwedge$ *p p′ q* . $[\![q \in HO\ p;\ rcvd\ p\ q \neq \lambda]\!] \Longrightarrow rcvd\ p'\ q \in \{rcvd\ p\ q,\ \lambda\}$ — no equivocation
    and *p4*:$\bigwedge$ *p* . $P{-}F \subseteq HO\ p$ — all participating, non-faulty processors are heard of
    and *p5*:$\bigwedge$ *p q* . $q \in C \Longrightarrow rcvd\ p\ q = bcast\ q$ — messages from participating, non-faulty processors are delivered intact
    and *p6*:$\bigwedge$ *p* . $HO\ p \subseteq P$ — only participating processors are heard of
    and *p7*:$\bigwedge$ *p* . $bcast\ p \neq \lambda$ — participating, non-faulty processors do not broadcast λ
    and *p8*:$\bigwedge$ *p p′ q* . $[\![q \in HO\ p;\ rcvd\ p\ q \neq \lambda]\!] \Longrightarrow q \in HO\ p'$ — if *p* receives a non-λ message form *q*, then all hear from *q*
begin

3 Properties of the Gafni-Losa model

lemma *maj-includes-correct*:
  — A majority among a heard-of set includes a correct processor
  fixes *M p*
  assumes *majority M* (*HO p*)
  obtains *q* **where** $q \in M \cap C$
proof −
  have *majority C* (*HO p*)
    by (*metis C-def card-mono finite majority-def order-le-less-trans p2 p4 p6*)
  thus *?thesis*
    using *majorities-intersect*
    by (*metis assms card.empty ex-in-conv less-nat-zero-code majority-def mult-0-right subset-empty that*)
qed

lemma *maj-not-lambda*:
  — If p hears of *m* from a majority, then $m \neq \lambda$
  fixes *p M m p′*
  assumes $M \subseteq HO\ p$
    and $\bigwedge$ *q* . $q \in M \Longrightarrow rcvd\ p\ q = m$
  shows *majority M* (*HO p*) $\Longrightarrow m \neq \lambda$
  by (*metis Int-iff assms*(*2*) *maj-includes-correct p5 p7*)

lemma *ho-sets-intersect*:
  fixes *p p′*
  shows $HO\ p \cap HO\ p' \neq \{\}$
  by (*metis C-def card.empty inf.absorb-iff2 inf-assoc inf-bot-left less-nat-zero-code majority-def mult-0-right p2 p4*)

lemma *maj-is-maj-among-hos*:
— If p receives m unanimously from a majority M then M is a majority among the processors that both p and p' hear of
fixes $p\ M\ m\ p'$
assumes $M \subseteq HO\ p$
and $\bigwedge q\ .\ q \in M \Longrightarrow rcvd\ p\ q = m$
and $majority\ M\ (HO\ p)$
shows $majority\ M\ (HO\ p \cap HO\ p')$
proof −
have $M \cap (HO\ p - HO\ p') = \{\}$
proof −
have $m \neq \lambda$
using ‹*majority M (HO p)*› *assms(1,2) maj-not-lambda* **by** *auto*
moreover
have $rcvd\ p\ q = \lambda$ **if** $q \in HO\ p - HO\ p'$ **for** q
by (*metis Diff-iff p8 that*)
ultimately show *?thesis* **using** *assms(1,2)*
by *blast*
qed
thus *?thesis* **using** *majority-anti* ‹*majority M (HO p)*›
by (*metis Diff-Diff-Int Int-empty-left inf-aci(1)*)
qed

lemma *faulty-min-among-hos*:
— F is a minority among the intersection of two heard-of sets
fixes $p\ p'$
shows $\neg majority\ (F \cap HO\ p \cap HO\ p')\ (HO\ p \cap HO\ p')$
proof −
have $majority\ C\ (HO\ p \cap HO\ p')$
by (*smt (verit) C-def Diff-Compl Diff-disjoint inf.absorb-iff2 inf.orderE inf-left-commute majority-anti p2 p4 p6*)
thus *?thesis*
by (*metis C-def Diff-disjoint empty-subsetI inf.assoc inf.commute inf.orderE ho-sets-intersect majorities-intersect*)
qed

end

4 Correctness of the commit-adopt algorithm

context *round*
begin

lemma *ca-lemma*:
— This is the most important lemma, from which the correctness of the commit-adopt algorithm follows
fixes $p\ p'\ m\text{-}1\ m\ M\text{-}1\ M\text{-}1'\ M'$
assumes $m \neq m\text{-}1$ **and** $m \neq \lambda$

and $\bigwedge$ *p . bcast p* $\neq$ *m* — processors never send *m*
defines *M-1* $\equiv$ {*q* $\in$ *HO p . rcvd p q* = *m-1*}
assumes *majority M-1* (*HO p*) — *p* receives *m-1* from a strict majority of the processors that it hears of
defines *M-1'* $\equiv$ {*q* $\in$ *HO p' . rcvd p' q* = *m-1*}
and *M'* $\equiv$ {*q* $\in$ *HO p' . rcvd p' q* = *m*}
shows *card M'* < *card M-1'* — *p'* receives *m-1* more often than *m*
proof −
have *m-1* $\neq$ λ
by (*metis* (*mono-tags*, *lifting*) *CollectD M-1-def assms*(*5*) *maj-not-lambda majority-def*)
define *F'* **where** *F'* $\equiv$ *F* $\cap$ *HO p* $\cap$ *HO p'*
have *M-1* − *F'* $\subseteq$ *M-1'* **unfolding** *M-1-def M-1'-def F'-def*
by (*smt* (*verit*, *del-insts*) *Diff-iff IntI* ‹*m-1* $\neq$ λ› *mem-Collect-eq round-axioms round-def subsetD subsetI*)
moreover
have *M'* $\subseteq$ *F'* − *M-1*
unfolding *M'-def M-1-def F'-def*
by (*clarify*, *smt* (*verit*, *ccfv-threshold*) *C-def DiffI IntI* ‹*m-1* $\neq$ λ› *assms*(*1*−*3*) *insertE mem-Collect-eq p5 p6 p8 round.p3 round-axioms singletonD subsetD*)
moreover
have *card* (*F'* − *M-1*) < *card* (*M-1* − *F'*)
proof −
have *card F'* < *card M-1*
by (*metis F'-def assms*(*5*) *card-maj-gt-card-not-maj faulty-min-among-hos inf.idem inf-assoc inf-le1 inf-left-commute maj-increasing*)
thus *?thesis*
by (*simp add*: *card-less-sym-Diff*)
qed
ultimately
show *?thesis*
by (*meson card-mono finite not-less order-le-less-trans*)
qed

end

4.1 Additional properties

context *round*
begin

lemma *l2*:
— There cannot be two different unanimous majorities
fixes *p p' M m m' M'*
assumes $\bigwedge$ *q . q* $\in$ *M* $\Longrightarrow$ *rcvd p q* = *m*
and *majority M* (*HO p*) — p receive *m* from a strict majority of the processors it hears of
and $\bigwedge$ *q . q* $\in$ *M'* $\Longrightarrow$ *rcvd p' q* = *m'*
and *majority M'* (*HO p'*) — p' receive *m'* from a strict majority of the processors

it hears of
  shows $m = m'$
proof $-$
  obtain q **where** $q \in M \cap M'$
  proof $-$
    have *majority* M $(HO\ p\ \cap HO\ p')$
      by (*meson assms*(*1*) *assms*(*2*) *maj-is-maj-among-hos majority-def*)
    moreover
    have *majority* M' $(HO\ p \cap HO\ p')$
      using *assms*(*4*) *assms*(*3*) *maj-is-maj-among-hos majority-def*
      by (*metis inf-commute*)
    moreover have $HO\ p \cap HO\ p' \neq \{\}$
      by (*simp add*: *ho-sets-intersect*)
    ultimately
    obtain q **where** $q \in M \cap M'$
      by (*meson all-not-in-conv majorities-intersect*)
    thus *?thesis* **..**
  qed
  moreover have $m \neq \lambda$ **and** $m' \neq \lambda$
    by (*metis assms*(*1*) *assms*(*2*) *maj-not-lambda majority-def*, *metis assms*(*4*) *assms*(*3*) *maj-not-lambda majority-def*)
  moreover have $M \subseteq HO\ p$
    by (*meson assms*(*2*) *majority-def*)
  ultimately
  show $m = m'$
    by (*metis* (*full-types*) *Int-iff assms*(*1*) *assms*(*3*) *empty-iff insert-iff p3 subsetD*)
qed

lemma *not-lambda*:
  — one cannot receive λ from a correct processor
  fixes $p\ q\ m$
  assumes $q \in C$ **and** *rcvd* $p\ q = m$
  shows $m \neq \lambda$
  using *C-def assms*(*1*) *assms*(*2*) *p5 p7* **by** *auto*

end

end