ImplicitQuorums

giuli

December 9, 2018

Contents

theory *ImplicitQuorums*
  imports *Main*
begin

1 quorums

locale *quorums* =
  fixes *quorum* :: *'node set* ⇒ *bool*
    assumes *quorum-union*: $\bigwedge$ *Q Q'* . ⟦*quorum Q*; *quorum Q'*⟧ ⟹ *quorum* (*Q* ∪ *Q'*)
begin

abbreviation *quorum-of* **where**
  quorum-of p Q ≡ *quorum Q* ∧ *p* ∈ *Q*

definition *blocks* **where**
  blocks R p ≡ ∀ *Q* . *quorum-of p Q* ⟶ *Q* ∩ *R* ≠ {}

abbreviation *blocked* **where** *blocked R* ≡ {*p* . *blocks R p*}

lemma *blocked-blocked-eq-blocked*:
  blocks (*blocked R*) *q* = *blocks R q*
  unfolding *blocks-def* **by** *fastforce*

lemma *l1*:
  assumes *finite S* **and** $S \neq \{\}$ **and** $\bigwedge\ p\ .\ p \in S \Longrightarrow \exists\ Q\ .\ \textit{quorum-of}\ p\ Q \wedge Q \subseteq S$
  shows *quorum S*
    — This is trivial by the quorum union property but seems clumsy to prove in Isabelle/HOL
proof −
  obtain *f* **where** *quorum-of p* (*f p*) **and** $f\ p \subseteq S$ **if** $p \in S$ **for** *p* **using** *assms*(*3*) **by** (*auto*; *metis*)
  have $\bigcup\ \{f\ p \mid p\ .\ p \in S\} = S$
  proof −
    have $\forall\ p \in S\ .\ p \in f\ p \wedge f\ p \subseteq S$
      by (*simp add*: ‹$\bigwedge p.\ p \in S \Longrightarrow f\ p \subseteq S$› ‹$\bigwedge p.\ p \in S \Longrightarrow \textit{quorum-of}\ p\ (f\ p)$›)
    thus $\bigcup\ \{f\ p \mid p\ .\ p \in S\} = S$ **by** *auto*
  qed
  moreover
  have *quorum* ($\bigcup\ \{f\ p \mid p\ .\ p \in S\}$)
  proof −
   have *wf* $\{(X,\ Y).\ X \subset Y \wedge \textit{finite}\ Y\}$ **by** (*metis finite-psubset-def wf-finite-psubset*)
      — We are going to use well-founded induction
    moreover
    have $\forall\ p \in S\ .\ p \in f\ p \wedge \textit{quorum}\ (f\ p)$
      by (*simp add*: ‹$\bigwedge p.\ p \in S \Longrightarrow f\ p \subseteq S$› ‹$\bigwedge p.\ p \in S \Longrightarrow \textit{quorum-of}\ p\ (f\ p)$›)
    moreover note ‹$S \neq \{\}$› **and** ‹*finite S*›
    ultimately
    show *quorum* ($\bigcup\ \{f\ p \mid p\ .\ p \in S\}$)
    proof (*induct S rule*:*wf-induct-rule*)
      — Is this also called Noetherian induction?
      case (*less S*)
      obtain S' *x* **where** $S = \textit{insert}\ x\ S'$ **and** $S' \neq S$ **using** ‹$S \neq \{\}$› ‹*finite S*›
        by (*metis finite.cases insertI1 mk-disjoint-insert*)
      have $S' \subset S$ **using** ‹$S = \textit{insert}\ x\ S'$› ‹$S' \neq S$› **by** *auto*
      moreover have $\forall\ p \in S'\ .\ p \in f\ p \wedge \textit{quorum}\ (f\ p)$
        by (*simp add*: ‹$\forall\ p \in S\ .\ p \in f\ p \wedge \textit{quorum}\ (f\ p)$› ‹$S = \textit{insert}\ x\ S'$›)
      moreover have *finite* S'
        using ‹*finite S*› ‹$S = \textit{insert}\ x\ S'$› **by** *auto*
      moreover note ‹*finite S*› *less.hyps*
       ultimately have *quorum* ($\bigcup\{f\ p \mid p\ .\ p \in S'\}$) **if** $S' \neq \{\}$ **using** *that* **by** *auto*
      moreover have $\{f\ p \mid p\ .\ p \in S\} = \textit{insert}\ (f\ x)\ \{f\ p \mid p\ .\ p \in S'\}$
        using ‹$S = \textit{insert}\ x\ S'$› **by** *auto*
      moreover have *quorum* (*f x*)
        by (*simp add*: ‹$\forall\ p \in S\ .\ p \in f\ p \wedge \textit{quorum}\ (f\ p)$› ‹$S = \textit{insert}\ x\ S'$›)
      ultimately show *?case* **using** *quorum-union*
        by (*cases* $S' = \{\}$, *auto*)
    qed
  qed
  ultimately show *?thesis* **by** *simp*

qed

end

1.1 Intact sets

locale *wb* = *quorums quorum* **for** *quorum* :: *'node set* ⇒ *bool* +
  fixes *W*::*'node set*
begin

abbreviation *B* **where** $B \equiv -W$

definition *is-intact* **where**
  $\textit{is-intact } I \equiv I \subseteq W \wedge \textit{quorum } I$
    $\wedge (\forall\ Q\ Q'\ .\ \textit{quorum } Q \wedge \textit{quorum } Q' \wedge Q \cap I \neq \{\} \wedge Q' \cap I \neq \{\} \longrightarrow W \cap Q \cap Q' \neq \{\})$

lemma *intact-union*:
  assumes *is-intact* I_1 **and** *is-intact* I_2 **and** $I_1 \cap I_2 \neq \{\}$
  shows *is-intact* $(I_1 \cup I_2)$
proof −
  have $I_1 \cup I_2 \subseteq W$
    using *assms(1) assms(2) is-intact-def* **by** *auto*
  moreover
  have *quorum* $(I_1 \cup I_2)$
    using ‹*is-intact* I_1› ‹*is-intact* I_2› **unfolding** *is-intact-def* **using** *quorum-union* **by** *auto*
  moreover
  have $W \cap Q_1 \cap Q_2 \neq \{\}$
    if *quorum* Q_1 **and** *quorum* Q_2 **and** $Q_1 \cap (I_1 \cup I_2) \neq \{\}$ **and** $Q_2 \cap (I_1 \cup I_2) \neq \{\}$
    for Q_1 Q_2
  proof −
    have $W \cap Q_1 \cap Q_2 \neq \{\}$ **if** $Q_1 \cap I \neq \{\}$ **and** $Q_2 \cap I \neq \{\}$ **and** $I = I_1 \vee I = I_2$ **for** I
      using ‹*is-intact* I_1› ‹*is-intact* I_2› ‹*quorum* Q_1› ‹*quorum* Q_2›
      by (*metis* ‹*is-intact* I_1› ‹*is-intact* I_2› *is-intact-def that*)
    moreover
    have ‹$W \cap Q_1 \cap Q_2 \neq \{\}$› **if** *is-intact* I_1 **and** *is-intact* I_2
      and $I_1 \cap I_2 \neq \{\}$ **and** $Q_1 \cap I_1 \neq \{\}$ **and** $Q_2 \cap I_2 \neq \{\}$
    for I_1 I_2 — We generalize to avoid repeating the argument twice
    proof −
      note ‹$I_1 \cap I_2 \neq \{\}$›
      moreover have *quorum* I_2 **using** ‹*is-intact* I_2› **unfolding** *is-intact-def* **by** *auto*
      ultimately have $I_2 \cap Q_1 \neq \{\}$ **using** ‹*is-intact* I_1› ‹*quorum* Q_1› ‹$Q_1 \cap I_1 \neq \{\}$›
        unfolding *is-intact-def* **using** *inf-sup-aci(1)* **by** *blast*
      thus $W \cap Q_1 \cap Q_2 \neq \{\}$ **using** ‹*is-intact* I_2› ‹*quorum* Q_2› ‹*quorum* Q_1› ‹Q_2

$\cap\ I_2 \neq \{\}$›
        unfolding *is-intact-def* **by** *blast*
    qed
    ultimately show *?thesis* **using** *assms that* **by** *auto*
  qed
  ultimately show *?thesis* **using** *assms*
    unfolding *is-intact-def* **by** *simp*
qed

1.2 The live set

definition L **where** $L \equiv W - (\mathit{blocked}\ B)$

lemma *l2*: $p \in L \Longrightarrow \exists\ Q \subseteq W.\ \mathit{quorum\text{-}of}\ p\ Q$
  unfolding *L-def blocks-def* **using** *DiffD2* **by** *auto*

lemma *l3*:
  assumes $p \in L$ **shows** $\exists\ Q \subseteq L\ .\ \mathit{quorum\text{-}of}\ p\ Q$
proof −
  have *False* **if** $\bigwedge\ Q\ .\ \mathit{quorum\text{-}of}\ p\ Q \Longrightarrow Q \cap (-L) \neq \{\}$
  proof −
    obtain Q **where** $\mathit{quorum\text{-}of}\ p\ Q$ **and** $Q \subseteq W$
      using *l2* ‹$p \in L$› **by** *auto*
    moreover have $Q \cap (-L) \neq \{\}$
      using *that* ‹$\mathit{quorum\text{-}of}\ p\ Q$› **by** *simp*
    ultimately show *False* **unfolding** *L-def blocks-def* **by** *auto*
  qed
  thus *?thesis*
    by *fastforce*
qed

lemma *l4*:
  assumes $L \neq \{\}$ **and** *finite* L
  shows *quorum* L **using** *l1 l3 assms* **by** *metis*

lemma *l5*: $\mathit{quorum}\ L' \Longrightarrow L' \subseteq W \Longrightarrow L' \subseteq L$
   unfolding *L-def blocks-def* **by** *auto*

lemma *l6*: $\mathit{is\text{-}intact}\ I \Longrightarrow I \neq \{\} \Longrightarrow I \subseteq L$
  by (*simp add*: *is-intact-def l5*)

end

2 Stellar quorums

locale *stellar* =
  fixes *slices* :: $'\mathit{node} \Rightarrow {'\mathit{node}}\ \mathit{set}\ \mathit{set}$ — the quorum slices
    and W :: $'\mathit{node}\ \mathit{set}$ — The well-behaved nodes
begin

definition *quorum* **where**
$quorum\ Q \equiv \forall\ p \in Q \cap W\ .\ \exists\ Sl \in slices\ p\ .\ Sl \subseteq Q$

lemma *quorum-union*:$quorum\ Q \Longrightarrow quorum\ Q' \Longrightarrow quorum\ (Q \cup Q')$
unfolding *quorum-def*
by (*metis IntE Int-iff UnE inf-sup-aci*(*1*) *sup.coboundedI1 sup.coboundedI2*)

interpretation *wb quorum W* **using** *quorum-union* **unfolding** *wb-def quorums-def*
by *auto*

lemma *quorum-is-quorum-of-some-slice*:
assumes *quorum-of* $p\ Q$ **and** $p \in W$
obtains S **where** $S \in slices\ p$ **and** $S \subseteq Q$
and $\bigwedge\ p'\ .\ p' \in S \cap W \Longrightarrow quorum\text{-}of\ p'\ Q$
using *assms* **unfolding** *quorum-def* **by** (*metis IntD1 Int-iff subsetCE*)

2.1 Inductive definition of blocked

inductive *blocking* **where**
$p \in R \Longrightarrow blocking\ R\ p$
$|\ \forall\ Sl \in slices\ p\ .\ \exists\ q \in Sl\ .\ blocking\ R\ q \Longrightarrow blocking\ R\ p$

~~inductive blocking2 where p ∈ R∪B ⟹ blocking2 R p| ∀ Sl ∈ slices p . ∃ q ∈ Sl . blocking2 R q ⟹ blocking2 R plemma p ∈ W ⟹ R ≠ {} ⟹ p ∉ R ⟹ blocking2 R p ⟹ blocks R p nitpick[card 'node=5, verbose, iter stellar.blocking2=5]~~

2.1.1 Properties of *blocking*

Here we show two main lemmas:

- if $R \cup B$ blocks $p \in Intact$, then $R \cap Intact \neq \{\}$
- if $p \in Intact$ and quorum Q is such that $Q \cap Intact \neq \{\}$, then $Q \cap W$ is blocking p

lemma *l7*:
assumes $blocking\ (R \cup B)\ p$ **and** $p \in W$
shows $blocks\ (R \cup B)\ p$
using *assms* **thm** *blocking.induct*
proof (*induct* $R \cup B\ p$ *rule*:*blocking.induct*)
case (*1 p*)
then show *?case*
using *blocks-def* **by** *auto*
next
case (*2 p*)
then show *?case* **unfolding** *blocks-def quorum-def*
by (*metis Compl-partition IntE Int-Un-distrib inf-sup-absorb subsetCE subset-refl sup-assoc sup-bot.left-neutral*)

```
qed

lemma l8:
  assumes is-intact I and p ∈ I and blocking (R ∪ B) p
  shows R ∩ I ≠ {}
proof −
  have quorum I and I ⊆ W and I ≠ {}
    using assms(1) is-intact-def using assms(2) by auto
  have blocks (R ∪ B) p using l7[OF ‹blocking (R ∪ B) p›] using ‹I ⊆ W› ‹p ∈
I› by auto
  hence (R ∪ B) ∩ Q ≠ {} if quorum-of p Q for Q
    using  blocks-def that by auto
  moreover
  have B ∩ I = {}
    using ComplD Int-emptyI ‹I ⊆ W› by auto
  moreover
  have quorum-of p I by (simp add: ‹quorum I› ‹p ∈ I›)
  ultimately
  show ?thesis
    by (metis Un-absorb assms(2) inf-sup-distrib2)
qed

inductive not-blocked for p R where
  ⟦p ∉ R; Sl ∈ slices p; ∀ q ∈ Sl . ¬ blocking R q⟧ ⟹ not-blocked p R p
| ⟦not-blocked p R p'; Sl ∈ slices p'; ∀ q ∈ Sl . ¬ blocking R q; p'' ∈ Sl⟧ ⟹
not-blocked p R p''

lemma not-blocked-self:not-blocked p R q ⟹ not-blocked p R p
proof (induct rule:not-blocked.induct)
  case (1 Sl)
  then show ?case
    using not-blocked.intros(1) by blast
next
  case (2 p' Sl p'')
  then show ?case
    by simp
qed

lemma l9:
  fixes Q p R
  defines Q ≡ {q . not-blocked p R q}
  shows quorum Q
proof −
  have ∀ n∈Q . ∃ S∈slices n . S⊆Q
  proof (simp add: Q-def; clarify)
    fix n
    assume not-blocked p R n
    thus ∃S∈slices n. S ⊆ Collect (not-blocked p R)
    proof (cases)
```

```
      case (1 Sl)
      then show ?thesis
        by (metis (full-types) Ball-Collect not-blocked.intros)
    next
      case (2 p′ Sl)
      hence ¬blocking R n by simp
       with this obtain Sl where n ∉ R and Sl ∈ slices n and ∀ q ∈ Sl . ¬
blocking R q
        by (meson blocking.intros(2) blocking.intros(1))
      moreover note ‹not-blocked p R n›
    ultimately show ?thesis by (metis (full-types) Ball-Collect not-blocked.intros(2))
    qed
  qed
  thus ?thesis by (simp add: quorum-def)
qed

lemma l10:
  fixes Q p R
  defines Q ≡ {q . not-blocked p R q}
  shows Q ∩ R = {}
proof −
  have q ∉ R if not-blocked p R q for q
    using that
  proof (induct)
    case (1 Sl)
    then show ?case by auto
  next
    case (2 p′ Sl p′′)
    then show ?case using blocking.intros(1) by blast
  qed
  thus ?thesis unfolding Q-def by auto
qed


lemma l11:
  assumes p ∈ W and blocks R p
  shows blocking R p
proof −
  define Q where Q ≡ {q . not-blocked p R q}
  have Q ≠ {} if ¬blocking R p unfolding Q-def
   by (metis blocking.intros(2) empty-Collect-eq not-blocked.intros(1) blocking.intros(1)
that)
  hence p ∈ Q if ¬blocking R p unfolding Q-def using not-blocked-self that by
blast
  moreover
  have quorum Q using l9 quorum-def Q-def by auto
  moreover have Q ∩ R = {} by (simp add: l10 Q-def)
  ultimately have ¬blocks R p if ¬blocking R p using that unfolding blocks-def
```

by *auto*
thus *?thesis* **using** ⟨*blocks R p*⟩
by *blast*
qed

lemma *l12*:
assumes *is-intact I* **and** $p \in I$ **and** $Q \cap I \neq \{\}$ **and** *quorum Q*
shows *blocking* $(Q \cap W)$ *p*
proof −
have *blocks* $(Q \cap W)$ *p* **using** *assms* **unfolding** *blocks-def is-intact-def*
using *disjoint-iff-not-equal* **by** *blast*
moreover have $p \in W$
using *assms*(*1*,*2*) *is-intact-def* **by** *auto*
ultimately
show *?thesis* **using** *l11* **by** *auto*
qed

3 Reachable part of a quorum

Here we define the part of a quorum Q of p that is reachable through well-behaved nodes from p. We show that if p and p' are intact and Q is a quorum of p and Q' is a quorum of p', then the intersection of Q, Q', and W is reachable from both p and p' through intact participants.

inductive *reachable* **for** *p Q* **where**
reachable p Q p
| $[\![$*reachable p Q p*$'$; $p' \in W$; $S \in$ *slices* p'; $S \subseteq Q$; $p'' \in S]\!] \Longrightarrow$ *reachable p Q* p''

definition *truncation* **where** *truncation p Q* $\equiv \{p'$. *reachable p Q* $p'\}$

lemma *l13*:
assumes *quorum Q* **and** $p \in Q \cap W$ **and** *reachable p Q* p'
shows $p' \in Q$
using *assms* **by** (*metis IntE contra-subsetD reachable.cases*)

lemma *l14*:
assumes *quorum Q* **and** $p \in Q \cap W$
shows *quorum* (*truncation p Q*)
proof −
have $\exists\ S \in$ *slices* p' . $\forall\ q \in S$. *reachable p Q q* **if** *reachable p Q* p' **and** $p' \in W$ **for** p'
by (*metis IntI assms l13 quorum-def stellar.reachable.simps that*)
thus *?thesis*
by (*metis IntE mem-Collect-eq stellar.quorum-def subsetI truncation-def*)
qed

lemma *l15*:
assumes *is-intact I* **and** *quorum Q* **and** *quorum* Q' **and** $p \in Q \cap I$ **and** $p' \in Q' \cap I$ **and** $Q \cap Q' \cap W \neq \{\}$

shows $W \cap (truncation\ p\ Q) \cap (truncation\ p'\ Q') \neq \{\}$
proof −
have *quorum* (*truncation p Q*) **and** *quorum* (*truncation p′ Q′*) **using** *l14 assms is-intact-def* **by** *auto*
moreover have *truncation p Q* $\cap I \neq \{\}$ **and** *truncation p′ Q′* $\cap I \neq \{\}$
by (*metis IntD2 Int-Collect assms*(*4*,*5*) *empty-iff inf-commute reachable.intros*(*1*) *stellar.truncation-def*)+
moreover note ⟨*is-intact I*⟩
ultimately show *?thesis* **unfolding** *is-intact-def* **by** *auto*
qed

end

4 elementary quorums

locale *elementary* = *stellar*
begin

definition *elementary* **where**
elementary s $\equiv$ *quorum s* $\wedge$ ($\forall$ *s′* . *s′* $\subset$ *s* $\longrightarrow$ $\neg$*quorum s′*)

lemma *finite-subset-wf*:
shows *wf* $\{(X,\ Y).\ X \subset Y \wedge$ *finite* $Y\}$
by (*metis finite-psubset-def wf-finite-psubset*)

lemma *quorum-contains-elementary*:
assumes *finite s* **and** $\neg$ *elementary s* **and** *quorum s*
shows $\exists$ *s′* . *s′* $\subset$ *s* $\wedge$ *elementary s′* **using** *assms*
proof (*induct s rule*:*wf-induct*[**where** *?r*=$\{(X,\ Y).\ X \subset Y \wedge$ *finite* $Y\}$])
case *1*
then show *?case* **using** *finite-subset-wf* **by** *simp*
next
case (*2 x*)
then show *?case*
by (*metis* (*full-types*) *elementary-def finite-psubset-def finite-subset in-finite-psubset less-le psubset-trans*)
qed

inductive *path* **where**
path []
| $\bigwedge$ *x* . *path* [*x*]
| $\bigwedge$ *l n* . ⟦*path l*; $S \in Q$ (*hd l*); $n \in S$⟧ $\Longrightarrow$ *path* (*n#l*)

lemma *elementary-connected*:
assumes *elementary s* **and** $n_1 \in s$ **and** $n_2 \in s$ **and** $n_1 \in W$ **and** $n_2 \in W$
shows $\exists$ *l* . *hd* (*rev l*) $= n_1 \wedge$ *hd l* $= n_2 \wedge$ *path l* (**is** *?P*)
proof −
{ assume $\neg$*?P*
define *x* **where** $x \equiv \{n \in s\ .\ \exists\ l\ .\ l \neq [] \wedge$ *hd* (*rev l*) $= n_1 \wedge$ *hd l* $= n \wedge$ *path*

l}
have $n_2 \notin x$ **using** ‹¬*?P*› *x-def* **by** *auto*
have $n_1 \in x$ **unfolding** *x-def* **using** *assms*(*2*) *path.intros*(*2*) **by** *force*
have *quorum x*
proof −
{ **fix** *n*
assume $n \in x$
have $\exists\ S\ .\ S \in$ *slices* $n \wedge S \subseteq x$
proof −
obtain *S* **where** $S \in$ *slices n* **and** $S \subseteq s$ **using** ‹*elementary s*› ‹$n \in x$›
unfolding *x-def*
by (*force simp add*:*elementary-def quorum-def*)
have $S \subseteq x$
proof −
{ **assume** $\neg\ S \subseteq x$
obtain *m* **where** $m \in S$ **and** $m \notin x$ **using** ‹$\neg\ S \subseteq x$› **by** *auto*
obtain l' **where** *hd* (*rev* l') $= n_1$ **and** *hd* $l' = n$ **and** *path* l' **and** $l' \neq$
[]
using ‹$n \in x$› *x-def* **by** *blast*
have *path* (m # l') **using** ‹*path* l'› ‹$m \in S$› ‹$S \in$ *slices n*› ‹*hd* $l' = n$›
using *path.intros*(*3*) **by** *fastforce*
moreover have *hd* (*rev* (m # l')) $= n_1$ **using** ‹*hd* (*rev* l') $= n_1$› ‹l'
$\neq$ []› **by** *auto*
ultimately have $m \in x$ **using** ‹$m \in S$› ‹$S \subseteq s$› *x-def* **by** *auto*
hence *False* **using** ‹$m \notin x$› **by** *blast* }
thus *?thesis* **by** *blast*
qed
thus *?thesis*
using ‹$S \in$ *slices n*› **by** *blast*
qed }
thus *?thesis* **by** (*meson Int-iff quorum-def*)
qed
moreover have $x \subset s$
using ‹$n_2 \notin x$› *assms*(*3*) *x-def* **by** *blast*
ultimately have *False* **using** ‹*elementary s*›
using *elementary-def* **by** *auto*
}
thus *?P* **by** *blast*
qed

end

end