

# Constructing the Reals as Dedekind Cuts of Rationals

Jacques D. Fleuriot and Lawrence C. Paulson

March 24, 2023

## **Abstract**

The type of real numbers is constructed from the positive rationals using the method of Dedekind cuts. This development, briefly described in papers by the authors [1, 2], follows the textbook presentation by Gleason [3]. It's notable that the first formalisation of a significant piece of mathematics, by Jutting [4] in 1977, involved a similar construction.

## Contents

<b>1</b>	<b>The Reals as Dedekind Sections of Positive Rationals</b>	<b>3</b>
1.1	Dedekind cuts or sections . . . . .	3
1.2	Properties of Ordering . . . . .	6
1.3	Properties of Addition . . . . .	7
1.4	Properties of Multiplication . . . . .	8
1.5	Distribution of Multiplication across Addition . . . . .	11
1.6	Existence of Inverse, a Positive Real . . . . .	12
1.7	Gleason's Lemma 9-3.4, page 122 . . . . .	13
1.8	Gleason's Lemma 9-3.6 . . . . .	15
1.9	Existence of Inverse: Part 2 . . . . .	16
1.10	Subtraction for Positive Reals . . . . .	18
1.11	Completeness of type <i>preal</i> . . . . .	21
1.12	Defining the Reals from the Positive Reals . . . . .	22
1.13	Equivalence relation over positive reals . . . . .	23
1.14	Addition and Subtraction . . . . .	24
1.15	Multiplication . . . . .	24
1.16	Inverse and Division . . . . .	25
1.17	The Real Numbers form a Field . . . . .	26
1.18	The $\leq$ Ordering . . . . .	27
1.19	The Reals Form an Ordered Field . . . . .	28
1.20	Completeness of the reals . . . . .	30
1.21	Theorems About the Ordering . . . . .	31
1.22	Completeness of Positive Reals . . . . .	31
1.23	Completeness . . . . .	32
1.24	The Archimedean Property of the Reals . . . . .	34

**Remark.** This development was part of the Isabelle distribution from about 1999 to 2022. It has been transferred to the AFP, where it may be more useful.

# 1 The Reals as Dedekind Sections of Positive Rationals

Fundamentals of Abstract Analysis [Gleason, p. 121] provides some of the definitions.

```
theory Dedekind-Real
imports Complex-Main
begin
```

```
lemma add-eq-exists:  $\exists x. a+x = (b::'a::ab-group-add)$ 
by (rule-tac x=b-a in exI, simp)
```

## 1.1 Dedekind cuts or sections

**definition**

```
cut :: rat set  $\Rightarrow$  bool where
cut A  $\equiv$   $\{ \} \subset A \wedge A \subset \{0<..\}$   $\wedge$ 
 $(\forall y \in A. ((\forall z. 0 < z \wedge z < y \longrightarrow z \in A) \wedge (\exists u \in A. y < u)))$ 
```

**lemma** *cut-of-rat*:

```
assumes q:  $0 < q$  shows cut  $\{r::rat. 0 < r \wedge r < q\}$  (is cut ?A)
```

**proof** –

```
from q have pos:  $?A \subset \{0<..\}$  by force
```

```
have nonempty:  $\{ \} \subset ?A$ 
```

**proof**

```
show  $\{ \} \subseteq ?A$  by simp
```

```
show  $\{ \} \neq ?A$ 
```

```
using field-lbound-gt-zero q by auto
```

**qed**

```
show ?thesis
```

```
by (simp add: cut-def pos nonempty,  
blast dest: dense intro: order-less-trans)
```

**qed**

**typedef** *preal* = *Collect cut*

```
by (blast intro: cut-of-rat [OF zero-less-one])
```

**lemma** *Abs-preal-induct* [*induct type: preal*]:

```
 $(\bigwedge x. cut\ x \Longrightarrow P\ (Abs-preal\ x)) \Longrightarrow P\ x$ 
```

```
using Abs-preal-induct [of P x] by simp
```

**lemma** *cut-Rep-preal* [*simp*]: *cut* (*Rep-preal x*)

```
using Rep-preal [of x] by simp
```

**definition**

```
psup :: preal set  $\Rightarrow$  preal where
```

```
psup P = Abs-preal  $(\bigcup X \in P. Rep-preal\ X)$ 
```

**definition**

*add-set* :: [rat set, rat set]  $\Rightarrow$  rat set **where**  
*add-set* A B = {w.  $\exists x \in A. \exists y \in B. w = x + y$ }

**definition**

*diff-set* :: [rat set, rat set]  $\Rightarrow$  rat set **where**  
*diff-set* A B = {w.  $\exists x. 0 < w \wedge 0 < x \wedge x \notin B \wedge x + w \in A$ }

**definition**

*mult-set* :: [rat set, rat set]  $\Rightarrow$  rat set **where**  
*mult-set* A B = {w.  $\exists x \in A. \exists y \in B. w = x * y$ }

**definition**

*inverse-set* :: rat set  $\Rightarrow$  rat set **where**  
*inverse-set* A  $\equiv$  {x.  $\exists y. 0 < x \wedge x < y \wedge \text{inverse } y \notin A$ }

**instantiation** *preal* :: {ord, plus, minus, times, inverse, one}  
**begin**

**definition**

*preal-less-def*:  
 $r < s \equiv \text{Rep-preal } r < \text{Rep-preal } s$

**definition**

*preal-le-def*:  
 $r \leq s \equiv \text{Rep-preal } r \subseteq \text{Rep-preal } s$

**definition**

*preal-add-def*:  
 $r + s \equiv \text{Abs-preal } (\text{add-set } (\text{Rep-preal } r) (\text{Rep-preal } s))$

**definition**

*preal-diff-def*:  
 $r - s \equiv \text{Abs-preal } (\text{diff-set } (\text{Rep-preal } r) (\text{Rep-preal } s))$

**definition**

*preal-mult-def*:  
 $r * s \equiv \text{Abs-preal } (\text{mult-set } (\text{Rep-preal } r) (\text{Rep-preal } s))$

**definition**

*preal-inverse-def*:  
 $\text{inverse } r \equiv \text{Abs-preal } (\text{inverse-set } (\text{Rep-preal } r))$

**definition**  $r \text{ div } s = r * \text{inverse } (s::\text{preal})$

**definition**

*preal-one-def*:  
 $1 \equiv \text{Abs-preal } \{x. 0 < x \wedge x < 1\}$

**instance ..**

**end**

Reduces equality on abstractions to equality on representatives

**declare** *Abs-preal-inject* [*simp*]

**declare** *Abs-preal-inverse* [*simp*]

**lemma** *rat-mem-preal*:  $0 < q \implies \text{cut } \{r::\text{rat}. 0 < r \wedge r < q\}$

**by** (*simp add: cut-of-rat*)

**lemma** *preal-nonempty*:  $\text{cut } A \implies \exists x \in A. 0 < x$

**unfolding** *cut-def* [*abs-def*] **by** *blast*

**lemma** *preal-Ex-mem*:  $\text{cut } A \implies \exists x. x \in A$

**using** *preal-nonempty* **by** *blast*

**lemma** *preal-exists-bound*:  $\text{cut } A \implies \exists x. 0 < x \wedge x \notin A$

**using** *Dedekind-Real.cut-def* **by** *fastforce*

**lemma** *preal-exists-greater*:  $\llbracket \text{cut } A; y \in A \rrbracket \implies \exists u \in A. y < u$

**unfolding** *cut-def* [*abs-def*] **by** *blast*

**lemma** *preal-downwards-closed*:  $\llbracket \text{cut } A; y \in A; 0 < z; z < y \rrbracket \implies z \in A$

**unfolding** *cut-def* [*abs-def*] **by** *blast*

Relaxing the final premise

**lemma** *preal-downwards-closed'*:  $\llbracket \text{cut } A; y \in A; 0 < z; z \leq y \rrbracket \implies z \in A$

**using** *less-eq-rat-def preal-downwards-closed* **by** *blast*

A positive fraction not in a positive real is an upper bound. Gleason p. 122 - Remark (1)

**lemma** *not-in-preal-ub*:

**assumes** *A*:  $\text{cut } A$

**and** *notx*:  $x \notin A$

**and** *y*:  $y \in A$

**and** *pos*:  $0 < x$

**shows**  $y < x$

**proof** (*cases rule: linorder-cases*)

**assume**  $x < y$

**with** *notx* **show** *?thesis*

**by** (*simp add: preal-downwards-closed [OF A y] pos*)

**next**

**assume**  $x = y$

**with** *notx* **and** *y* **show** *?thesis* **by** *simp*

**next**

**assume**  $y < x$

**thus** *?thesis* .

**qed**

preal lemmas instantiated to *Rep-preal X*

**lemma** *mem-Rep-preal-Ex*:  $\exists x. x \in \text{Rep-preal } X$

**thm** *preal-Ex-mem*

**by** (rule *preal-Ex-mem* [OF *cut-Rep-preal*])

**lemma** *Rep-preal-exists-bound*:  $\exists x > 0. x \notin \text{Rep-preal } X$

**by** (rule *preal-exists-bound* [OF *cut-Rep-preal*])

**lemmas** *not-in-Rep-preal-ub = not-in-preal-ub* [OF *cut-Rep-preal*]

## 1.2 Properties of Ordering

**instance** *preal* :: *order*

**proof**

**fix** *w* :: *preal*

**show**  $w \leq w$  **by** (*simp add: preal-le-def*)

**next**

**fix** *i j k* :: *preal*

**assume**  $i \leq j$  **and**  $j \leq k$

**then show**  $i \leq k$  **by** (*simp add: preal-le-def*)

**next**

**fix** *z w* :: *preal*

**assume**  $z \leq w$  **and**  $w \leq z$

**then show**  $z = w$  **by** (*simp add: preal-le-def Rep-preal-inject*)

**next**

**fix** *z w* :: *preal*

**show**  $z < w \longleftrightarrow z \leq w \wedge \neg w \leq z$

**by** (*auto simp: preal-le-def preal-less-def Rep-preal-inject*)

**qed**

**lemma** *preal-imp-pos*:  $\llbracket \text{cut } A; r \in A \rrbracket \implies 0 < r$

**by** (*auto simp: cut-def*)

**instance** *preal* :: *linorder*

**proof**

**fix** *x y* :: *preal*

**show**  $x \leq y \vee y \leq x$

**unfolding** *preal-le-def*

**by** (*meson cut-Rep-preal not-in-preal-ub preal-downwards-closed preal-imp-pos subsetI*)

**qed**

**instantiation** *preal* :: *distrib-lattice*

**begin**

**definition**

(*inf* :: *preal*  $\Rightarrow$  *preal*  $\Rightarrow$  *preal*) = *min*

**definition**

$(sup :: preal \Rightarrow preal \Rightarrow preal) = max$

**instance**

**by** *intro-classes*

(*auto simp: inf-preal-def sup-preal-def max-min-distrib2*)

**end**

### 1.3 Properties of Addition

**lemma** *preal-add-commute*:  $(x::preal) + y = y + x$

**unfolding** *preal-add-def add-set-def*

**by** (*metis (no-types, opaque-lifting) add.commute*)

Lemmas for proving that addition of two positive reals gives a positive real

**lemma** *mem-add-set*:

**assumes** *cut A cut B*

**shows** *cut (add-set A B)*

**proof** –

**have**  $\{\} \subset add-set A B$

**using** *assms* **by** (*force simp: add-set-def dest: preal-nonempty*)

**moreover**

**obtain** *q* **where**  $q > 0 \ q \notin add-set A B$

**proof** –

**obtain** *a b* **where**  $a > 0 \ a \notin A \ b > 0 \ b \notin B \ \wedge x. x \in A \implies x < a \ \wedge y. y \in B \implies y < b$

**by** (*meson assms preal-exists-bound not-in-preal-ub*)

**with** *assms* **have**  $a+b \notin add-set A B$

**by** (*fastforce simp add: add-set-def*)

**then show** *thesis*

**using**  $\langle 0 < a \rangle \langle 0 < b \rangle$  *add-pos-pos* **that** **by** *blast*

**qed**

**then have**  $add-set A B \subset \{0<..\}$

**unfolding** *add-set-def*

**using** *preal-imp-pos [OF <cut A>] preal-imp-pos [OF <cut B>]* **by** *fastforce*

**moreover have**  $z \in add-set A B$

**if**  $u: u \in add-set A B$  **and**  $0 < z \ z < u$  **for**  $u \ z$

**using**  $u$  **unfolding** *add-set-def*

**proof** (*clarify*)

**fix**  $x::rat$  **and**  $y::rat$

**assume**  $ueq: u = x + y$  **and**  $x: x \in A$  **and**  $y: y \in B$

**have**  $xpos [simp]: x > 0$  **and**  $ypos [simp]: y > 0$

**using** *assms preal-imp-pos x y* **by** *blast+*

**have**  $xypos [simp]: x+y > 0$  **by** (*simp add: pos-add-strict*)

**let**  $?f = z/(x+y)$

**have**  $fless: ?f < 1$

**using** *divide-less-eq-1-pos <z < u> ueq xypos* **by** *blast*

**show**  $\exists x' \in A. \exists y' \in B. z = x' + y'$

```

proof (intro be $\lambda$ I)
  show  $z = x * ?f + y * ?f$ 
  by (simp add: distrib-right [symmetric] divide-inverse ac-simps order-less-imp-not-eq2)
next
  show  $y * ?f \in B$ 
  proof (rule preal-downwards-closed [OF  $\langle$ cut B $\rangle$  y])
    show  $0 < y * ?f$ 
    by (simp add:  $\langle$ 0 < z $\rangle$ )
  next
    show  $y * ?f < y$ 
    by (insert mult-strict-left-mono [OF fless ypos], simp)
  qed
next
  show  $x * ?f \in A$ 
  proof (rule preal-downwards-closed [OF  $\langle$ cut A $\rangle$  x])
    show  $0 < x * ?f$ 
    by (simp add:  $\langle$ 0 < z $\rangle$ )
  next
    show  $x * ?f < x$ 
    by (insert mult-strict-left-mono [OF fless xpos], simp)
  qed
qed
moreover
  have  $\bigwedge y. y \in \text{add-set } A \ B \implies \exists u \in \text{add-set } A \ B. y < u$ 
  unfolding add-set-def using preal-exists-greater assms by fastforce
  ultimately show ?thesis
  by (simp add: Dedekind-Real.cut-def)
qed

lemma preal-add-assoc:  $((x::\text{preal}) + y) + z = x + (y + z)$ 
apply (simp add: preal-add-def mem-add-set)
apply (force simp: add-set-def ac-simps)
done

```

```

instance preal :: ab-semigroup-add
proof
  fix a b c :: preal
  show  $(a + b) + c = a + (b + c)$  by (rule preal-add-assoc)
  show  $a + b = b + a$  by (rule preal-add-commute)
qed

```

## 1.4 Properties of Multiplication

Proofs essentially same as for addition

```

lemma preal-mult-commute:  $(x::\text{preal}) * y = y * x$ 
unfolding preal-mult-def mult-set-def
by (metis (no-types, opaque-lifting) mult.commute)

```

Multiplication of two positive reals gives a positive real.



```

lemma mem-mult-set:
  assumes cut A cut B
  shows cut (mult-set A B)
proof -
  have {}  $\subset$  mult-set A B
  using assms
  by (force simp: mult-set-def dest: preal-nonempty)
  moreover
  obtain q where q > 0 q  $\notin$  mult-set A B
  proof -
    obtain x y where x [simp]: 0 < x x  $\notin$  A and y [simp]: 0 < y y  $\notin$  B
    using preal-exists-bound assms by blast
    show thesis
    proof
      show 0 < x*y by simp
      show x * y  $\notin$  mult-set A B
      proof -
        {
          fix u::rat and v::rat
          assume u: u  $\in$  A and v: v  $\in$  B and xy: x*y = u*v
          moreover have u < x and v < y using assms x y u v by (blast dest:
not-in-preal-ub)+
          moreover have 0  $\leq$  v
            using less-imp-le preal-imp-pos assms x y u v by blast
          moreover have u*v < x*y
            using assms x < u < x < v < y < 0  $\leq$  v by (blast intro: mult-strict-mono)
          ultimately have False by force
        }
      thus ?thesis by (auto simp: mult-set-def)
    qed
  qed
  then have mult-set A B  $\subset$  {0<..}
  unfolding mult-set-def
  using preal-imp-pos [OF <cut A>] preal-imp-pos [OF <cut B>] by fastforce
  moreover have z  $\in$  mult-set A B
  if u: u  $\in$  mult-set A B and 0 < z z < u for u z
  using u unfolding mult-set-def
  proof (clarify)
    fix x::rat and y::rat
    assume ueq: u = x * y and x: x  $\in$  A and y: y  $\in$  B
    have [simp]: y > 0
      using <cut B> preal-imp-pos y by blast
    show  $\exists x' \in A. \exists y' \in B. z = x' * y'$ 
    proof
      have z = (z/y)*y
        by (simp add: divide-inverse mult.commute [of y] mult.assoc order-less-imp-not-eq2)
      then show  $\exists y' \in B. z = (z/y) * y'$ 
        using y by blast
    qed
  qed

```

```

next
  show  $z/y \in A$ 
  proof (rule preal-downwards-closed [OF ‹cut A› x])
    show  $0 < z/y$ 
    by (simp add: ‹0 < z›)
    show  $z/y < x$ 
    using ‹0 < y› pos-divide-less-eq ‹z < u› ueq by blast
  qed
qed
qed
moreover have  $\bigwedge y. y \in \text{mult-set } A \ B \implies \exists u \in \text{mult-set } A \ B. y < u$ 
  apply (simp add: mult-set-def)
  by (metis preal-exists-greater mult-strict-right-mono preal-imp-pos assms)
ultimately show ?thesis
  by (simp add: Dedekind-Real.cut-def)
qed

lemma preal-mult-assoc:  $((x::\text{preal}) * y) * z = x * (y * z)$ 
  apply (simp add: preal-mult-def mem-mult-set Rep-preal)
  apply (simp add: mult-set-def)
  apply (metis (no-types, opaque-lifting) ab-semigroup-mult-class.mult-ac(1))
  done

instance preal :: ab-semigroup-mult
proof
  fix a b c :: preal
  show  $(a * b) * c = a * (b * c)$  by (rule preal-mult-assoc)
  show  $a * b = b * a$  by (rule preal-mult-commute)
qed

Positive real 1 is the multiplicative identity element

lemma preal-mult-1:  $(1::\text{preal}) * z = z$ 
proof (induct z)
  fix A :: rat set
  assume A: cut A
  have  $\{w. \exists u. 0 < u \wedge u < 1 \wedge (\exists v \in A. w = u * v)\} = A$  (is ?lhs = A)
  proof
    show  $?lhs \subseteq A$ 
    proof clarify
      fix x::rat and u::rat and v::rat
      assume upos:  $0 < u$  and u<1 and v:  $v \in A$ 
      have vpos:  $0 < v$  by (rule preal-imp-pos [OF A v])
      hence  $u*v < 1*v$  by (simp only: mult-strict-right-mono upos ‹u < 1› v)
      thus  $u * v \in A$ 
      by (force intro: preal-downwards-closed [OF A v] mult-pos-pos upos vpos)
    qed
  qed
next
  show  $A \subseteq ?lhs$ 
  proof clarify

```

```

fix x::rat
assume x: x ∈ A
have xpos: 0 < x by (rule preal-imp-pos [OF A x])
from preal-exists-greater [OF A x]
obtain v where v: v ∈ A and xlessv: x < v ..
have vpos: 0 < v by (rule preal-imp-pos [OF A v])
show ∃ u. 0 < u ∧ u < 1 ∧ (∃ v ∈ A. x = u * v)
proof (intro exI conjI)
  show 0 < x/v
    by (simp add: zero-less-divide-iff xpos vpos)
  show x / v < 1
    by (simp add: pos-divide-less-eq vpos xlessv)
  have x = (x/v)*v
    by (simp add: divide-inverse mult.assoc vpos order-less-imp-not-eq2)
  then show ∃ v' ∈ A. x = (x / v) * v'
    using v by blast
qed
qed
qed
thus 1 * Abs-preal A = Abs-preal A
  by (simp add: preal-one-def preal-mult-def mult-set-def rat-mem-preal A)
qed

```

```

instance preal :: comm-monoid-mult
  by intro-classes (rule preal-mult-1)

```

## 1.5 Distribution of Multiplication across Addition

```

lemma mem-Rep-preal-add-iff:
  (z ∈ Rep-preal(r+s)) = (∃ x ∈ Rep-preal r. ∃ y ∈ Rep-preal s. z = x + y)
  apply (simp add: preal-add-def mem-add-set Rep-preal)
  apply (simp add: add-set-def)
  done

```

```

lemma mem-Rep-preal-mult-iff:
  (z ∈ Rep-preal(r*s)) = (∃ x ∈ Rep-preal r. ∃ y ∈ Rep-preal s. z = x * y)
  apply (simp add: preal-mult-def mem-mult-set Rep-preal)
  apply (simp add: mult-set-def)
  done

```

```

lemma distrib-subset1:
  Rep-preal (w * (x + y)) ⊆ Rep-preal (w * x + w * y)
  by (force simp: Bex-def mem-Rep-preal-add-iff mem-Rep-preal-mult-iff distrib-left)

```

```

lemma preal-add-mult-distrib-mean:
  assumes a: a ∈ Rep-preal w
  and b: b ∈ Rep-preal w
  and d: d ∈ Rep-preal x
  and e: e ∈ Rep-preal y

```

**shows**  $\exists c \in \text{Rep-preal } w. a * d + b * e = c * (d + e)$   
**proof**  
**let**  $?c = (a*d + b*e)/(d+e)$   
**have**  $[simp]: 0 < a \ 0 < b \ 0 < d \ 0 < e \ 0 < d+e$   
**by**  $(blast \ intro: \text{preal-imp-pos} \ [OF \ \text{cut-Rep-preal}] \ a \ b \ d \ e \ \text{pos-add-strict})+$   
**have**  $cpos: 0 < ?c$   
**by**  $(simp \ add: \ \text{zero-less-divide-iff} \ \text{zero-less-mult-iff} \ \text{pos-add-strict})$   
**show**  $a * d + b * e = ?c * (d + e)$   
**by**  $(simp \ add: \ \text{divide-inverse} \ \text{mult.assoc} \ \text{order-less-imp-not-eq2})$   
**show**  $?c \in \text{Rep-preal } w$   
**proof**  $(cases \ rule: \ \text{linorder-le-cases})$   
**assume**  $a \leq b$   
**hence**  $?c \leq b$   
**by**  $(simp \ add: \ \text{pos-divide-le-eq} \ \text{distrib-left} \ \text{mult-right-mono} \ \text{order-less-imp-le})$   
**thus**  $?thesis$  **by**  $(rule \ \text{preal-downwards-closed}' \ [OF \ \text{cut-Rep-preal} \ b \ cpos])$   
**next**  
**assume**  $b \leq a$   
**hence**  $?c \leq a$   
**by**  $(simp \ add: \ \text{pos-divide-le-eq} \ \text{distrib-left} \ \text{mult-right-mono} \ \text{order-less-imp-le})$   
**thus**  $?thesis$  **by**  $(rule \ \text{preal-downwards-closed}' \ [OF \ \text{cut-Rep-preal} \ a \ cpos])$   
**qed**  
**qed**

**lemma** *distrib-subset2*:

$\text{Rep-preal } (w * x + w * y) \subseteq \text{Rep-preal } (w * (x + y))$

**apply**  $(clarsimp \ simp: \ \text{mem-Rep-preal-add-iff} \ \text{mem-Rep-preal-mult-iff})$

**using**  $\text{mem-Rep-preal-add-iff} \ \text{preal-add-mult-distrib-mean}$  **by** *blast*

**lemma** *preal-add-mult-distrib2*:  $(w * ((x::\text{preal}) + y)) = (w * x) + (w * y)$

**by**  $(metis \ \text{Rep-preal-inverse} \ \text{distrib-subset1} \ \text{distrib-subset2} \ \text{subset-antisym})$

**lemma** *preal-add-mult-distrib*:  $((x::\text{preal}) + y) * w = (x * w) + (y * w)$

**by**  $(simp \ add: \ \text{preal-mult-commute} \ \text{preal-add-mult-distrib2})$

**instance** *preal :: comm-semiring*

**by**  $\text{intro-classes} \ (rule \ \text{preal-add-mult-distrib})$

## 1.6 Existence of Inverse, a Positive Real

**lemma** *mem-inverse-set*:

**assumes** *cut A* **shows** *cut (inverse-set A)*

**proof** –

**have**  $\exists x \ y. 0 < x \wedge x < y \wedge \text{inverse } y \notin A$

**proof** –

**from** *preal-exists-bound*  $[OF \ \langle \text{cut } A \rangle]$

**obtain**  $x$  **where**  $[simp]: 0 < x \ x \notin A$  **by** *blast*

**show** *?thesis*

```

proof (intro exI conjI)
  show  $0 < \text{inverse}(x+1)$ 
    by (simp add: order-less-trans [OF - less-add-one])
  show  $\text{inverse}(x+1) < \text{inverse } x$ 
    by (simp add: less-imp-inverse-less less-add-one)
  show  $\text{inverse}(\text{inverse } x) \notin A$ 
    by (simp add: order-less-imp-not-eq2)
qed
qed
then have  $\{\} \subset \text{inverse-set } A$ 
  using inverse-set-def by fastforce
moreover obtain  $q$  where  $q > 0$   $q \notin \text{inverse-set } A$ 
proof -
  from preal-nonempty [OF <cut A>]
  obtain  $x$  where  $x: x \in A$  and  $xpos$  [simp]:  $0 < x$  ..
  show ?thesis
  proof
    show  $0 < \text{inverse } x$  by simp
    show  $\text{inverse } x \notin \text{inverse-set } A$ 
    proof -
      { fix  $y::\text{rat}$ 
        assume  $ygt: \text{inverse } x < y$ 
        have [simp]:  $0 < y$  by (simp add: order-less-trans [OF - ygt])
        have  $iyless: \text{inverse } y < x$ 
          by (simp add: inverse-less-imp-less [of x] ygt)
        have  $\text{inverse } y \in A$ 
          by (simp add: preal-downwards-closed [OF <cut A> x] iyless)}
      thus ?thesis by (auto simp: inverse-set-def)
    qed
  qed
qed
moreover have  $\text{inverse-set } A \subset \{0 < ..\}$ 
  using calculation inverse-set-def by blast
moreover have  $z \in \text{inverse-set } A$ 
  if  $u: u \in \text{inverse-set } A$  and  $0 < z < u$  for  $u$   $z$ 
  using  $u$  that less-trans unfolding inverse-set-def by auto
moreover have  $\bigwedge y. y \in \text{inverse-set } A \implies \exists u \in \text{inverse-set } A. y < u$ 
  by (simp add: inverse-set-def) (meson dense less-trans)
ultimately show ?thesis
  by (simp add: Dedekind-Real.cut-def)
qed

```

## 1.7 Gleason's Lemma 9-3.4, page 122

**lemma** *Gleason9-34-exists:*

```

assumes  $A: \text{cut } A$ 
  and  $\forall x \in A. x + u \in A$ 
  and  $0 \leq z$ 
shows  $\exists b \in A. b + (\text{of-int } z) * u \in A$ 

```

```

proof (cases z rule: int-cases)
  case (nonneg n)
  show ?thesis
  proof (simp add: nonneg, induct n)
    case 0
    from preal-nonempty [OF A]
    show ?case by force
  next
    case (Suc k)
    then obtain b where b: b ∈ A b + of-nat k * u ∈ A ..
    hence b + of-int (int k)*u + u ∈ A by (simp add: assms)
    thus ?case by (force simp: algebra-simps b)
  qed
next
  case (neg n)
  with assms show ?thesis by simp
qed

lemma Gleason9-34-contr:
  assumes A: cut A
  shows  $\llbracket \forall x \in A. x + u \in A; 0 < u; 0 < y; y \notin A \rrbracket \implies \text{False}$ 
proof (induct u, induct y)
  fix a::int and b::int
  fix c::int and d::int
  assume bpos [simp]: 0 < b
    and dpos [simp]: 0 < d
    and closed:  $\forall x \in A. x + (\text{Fract } c \ d) \in A$ 
    and upos: 0 < Fract c d
    and ypos: 0 < Fract a b
    and notin: Fract a b  $\notin$  A
  have cpos [simp]: 0 < c
    by (simp add: zero-less-Fract-iff [OF dpos, symmetric] upos)
  have apos [simp]: 0 < a
    by (simp add: zero-less-Fract-iff [OF bpos, symmetric] ypos)
  let ?k = a*d
  have frle: Fract a b  $\leq$  Fract ?k 1 * (Fract c d)
  proof –
    have ?thesis = ((a * d * b * d)  $\leq$  c * b * (a * d * b * d))
      by (simp add: order-less-imp-not-eq2 ac-simps)
    moreover
    have (1 * (a * d * b * d))  $\leq$  c * b * (a * d * b * d)
      by (rule mult-mono,
        simp-all add: int-one-le-iff-zero-less zero-less-mult-iff
          order-less-imp-le)
  ultimately
  show ?thesis by simp
qed
have k: 0  $\leq$  ?k by (simp add: order-less-imp-le zero-less-mult-iff)
from Gleason9-34-exists [OF A closed k]

```

**obtain**  $z$  **where**  $z: z \in A$   
     **and**  $mem: z + of-int ?k * Fract c d \in A ..$   
**have**  $less: z + of-int ?k * Fract c d < Fract a b$   
     **by** (rule not-in-preal-ub [OF A notin mem ypos])  
**have**  $0 < z$  **by** (rule preal-imp-pos [OF A z])  
**with**  $frle$  **and**  $less$  **show**  $False$  **by** (simp add: Fract-of-int-eq)  
**qed**

**lemma** *Gleason9-34*:  
 assumes  $cut A 0 < u$   
 shows  $\exists r \in A. r + u \notin A$   
 using *assms Gleason9-34-contr preal-exists-bound* **by** *blast*

## 1.8 Gleason's Lemma 9-3.6

**lemma** *lemma-gleason9-36*:  
 assumes  $A: cut A$   
     **and**  $x: 1 < x$   
 shows  $\exists r \in A. r * x \notin A$   
**proof** –  
 from *preal-nonempty* [OF A]  
**obtain**  $y$  **where**  $y: y \in A$  **and**  $ypos: 0 < y ..$   
**show** *?thesis*  
**proof** (rule *classical*)  
     **assume**  $\sim(\exists r \in A. r * x \notin A)$   
     **with**  $y$  **have**  $ymem: y * x \in A$  **by** *blast*  
     **from**  $ypos$  *mult-strict-left-mono* [OF x]  
     **have**  $yless: y < y * x$  **by** *simp*  
     **let**  $?d = y * x - y$   
     **from**  $yless$  **have**  $dpos: 0 < ?d$  **and**  $eq: y + ?d = y * x$  **by** *auto*  
     **from** *Gleason9-34* [OF A dpos]  
     **obtain**  $r$  **where**  $r: r \in A$  **and**  $notin: r + ?d \notin A ..$   
     **have**  $rpos: 0 < r$  **by** (rule *preal-imp-pos* [OF A r])  
     **with**  $dpos$  **have**  $rdpos: 0 < r + ?d$  **by** *arith*  
     **have**  $\sim(r + ?d \leq y + ?d)$   
     **proof**  
         **assume**  $le: r + ?d \leq y + ?d$   
         **from**  $ymem$  **have**  $yd: y + ?d \in A$  **by** (simp add: eq)  
         **have**  $r + ?d \in A$  **by** (rule *preal-downwards-closed'* [OF A yd rdpos le])  
         **with**  $notin$  **show**  $False$  **by** *simp*  
     **qed**  
     **hence**  $y < r$  **by** *simp*  
     **with**  $ypos$  **have**  $dless: ?d < (r * ?d) / y$   
         **using**  $dpos$  *less-divide-eq-1* **by** *fastforce*  
     **have**  $r + ?d < r * x$   
     **proof** –  
         **have**  $r + ?d < r + (r * ?d) / y$  **by** (simp add: dless)  
         **also from**  $ypos$  **have**  $\dots = (r / y) * (y + ?d)$

by (*simp only: algebra-simps divide-inverse, simp*)  
 also have  $\dots = r*x$  using *ypos*  
 by *simp*  
 finally show  $r + ?d < r*x$  .  
 qed  
 with  $r \text{ notin } rdpos$   
 show  $\exists r \in A. r * x \notin A$  by (*blast dest: preal-downwards-closed [OF A]*)  
 qed  
 qed

## 1.9 Existence of Inverse: Part 2

**lemma** *mem-Rep-preal-inverse-iff:*

$(z \in \text{Rep-preal}(\text{inverse } r)) \longleftrightarrow (0 < z \wedge (\exists y. z < y \wedge \text{inverse } y \notin \text{Rep-preal } r))$

**apply** (*simp add: preal-inverse-def mem-inverse-set Rep-preal*)

**apply** (*simp add: inverse-set-def*)

**done**

**lemma** *Rep-preal-one:*

$\text{Rep-preal } 1 = \{x. 0 < x \wedge x < 1\}$

**by** (*simp add: preal-one-def rat-mem-preal*)

**lemma** *subset-inverse-mult-lemma:*

**assumes** *xpos:  $0 < x$  and xless:  $x < 1$*

**shows**  $\exists v u y. 0 < v \wedge v < y \wedge \text{inverse } y \notin \text{Rep-preal } R \wedge$   
 $u \in \text{Rep-preal } R \wedge x = v * u$

**proof** –

**from** *xpos* and *xless* **have**  $1 < \text{inverse } x$  **by** (*simp add: one-less-inverse-iff*)

**from** *lemma-gleason9-36 [OF cut-Rep-preal this]*

**obtain**  $t$  **where**  $t \in \text{Rep-preal } R$

**and** *notin:  $t * (\text{inverse } x) \notin \text{Rep-preal } R$  ..*

**have** *rpos:  $0 < t$*  **by** (*rule preal-imp-pos [OF cut-Rep-preal t]*)

**from** *preal-exists-greater [OF cut-Rep-preal t]*

**obtain**  $u$  **where**  $u \in \text{Rep-preal } R$  **and** *rless:  $t < u$  ..*

**have** *upos:  $0 < u$*  **by** (*rule preal-imp-pos [OF cut-Rep-preal u]*)

**show** *?thesis*

**proof** (*intro exI conjI*)

**show**  $0 < x/u$  **using** *xpos upos*

**by** (*simp add: zero-less-divide-iff*)

**show**  $x/u < x/t$  **using** *xpos upos rpos*

**by** (*simp add: divide-inverse mult-less-cancel-left rless*)

**show**  $\text{inverse } (x / t) \notin \text{Rep-preal } R$  **using** *notin*

**by** (*simp add: divide-inverse mult.commute*)

**show**  $u \in \text{Rep-preal } R$  **by** (*rule u*)

**show**  $x = x / u * u$  **using** *upos*

**by** (*simp add: divide-inverse mult.commute*)

qed

qed



**lemma** *subset-inverse-mult:*

$Rep\text{-preal } 1 \subseteq Rep\text{-preal}(inverse\ r * r)$

**by** (*force simp: Rep-preal-one mem-Rep-preal-inverse-iff mem-Rep-preal-mult-iff*  
*dest: subset-inverse-mult-lemma*)

**lemma** *inverse-mult-subset:*  $Rep\text{-preal}(inverse\ r * r) \subseteq Rep\text{-preal } 1$

**proof** –

**have**  $0 < u * v$  **if**  $v \in Rep\text{-preal } r$   $0 < u$   $u < t$  **for**  $u\ v\ t :: rat$

**using** *that by (simp add: zero-less-mult-iff preal-imp-pos [OF cut-Rep-preal])*

**moreover have**  $t * q < 1$

**if**  $q \in Rep\text{-preal } r$   $0 < t$   $t < y$   $inverse\ y \notin Rep\text{-preal } r$

**for**  $t\ q\ y :: rat$

**proof** –

**have**  $q < inverse\ y$

**using** *not-in-Rep-preal-ub that by auto*

**hence**  $t * q < t/y$

**using** *that by (simp add: divide-inverse mult-less-cancel-left)*

**also have**  $\dots \leq 1$

**using** *that by (simp add: pos-divide-le-eq)*

**finally show** *?thesis* .

**qed**

**ultimately show** *?thesis*

**by** (*auto simp: Rep-preal-one mem-Rep-preal-inverse-iff mem-Rep-preal-mult-iff*)

**qed**

**lemma** *preal-mult-inverse:*  $inverse\ r * r = (1::preal)$

**by** (*meson Rep-preal-inject inverse-mult-subset subset-antisym subset-inverse-mult*)

**lemma** *preal-mult-inverse-right:*  $r * inverse\ r = (1::preal)$

**using** *preal-mult-commute preal-mult-inverse by auto*

Theorems needing *Gleason9-34*

**lemma** *Rep-preal-self-subset:*  $Rep\text{-preal } (r) \subseteq Rep\text{-preal}(r + s)$

**proof**

**fix**  $x$

**assume**  $x: x \in Rep\text{-preal } r$

**obtain**  $y$  **where**  $y: y \in Rep\text{-preal } s$  **and**  $y > 0$

**using** *Rep-preal preal-nonempty by blast*

**have**  $ry: x+y \in Rep\text{-preal}(r + s)$  **using**  $x\ y$

**by** (*auto simp: mem-Rep-preal-add-iff*)

**then show**  $x \in Rep\text{-preal}(r + s)$

**by** (*meson <0 < y> add-less-same-cancel1 not-in-Rep-preal-ub order.asym preal-imp-pos*  
*[OF cut-Rep-preal x]*)

**qed**

**lemma** *Rep-preal-sum-not-subset:*  $\sim Rep\text{-preal } (r + s) \subseteq Rep\text{-preal}(r)$

**proof** –

**obtain**  $y$  **where**  $y: y \in Rep\text{-preal } s$  **and**  $y > 0$

**using** *Rep-preal preal-nonempty by blast*

**obtain**  $x$  **where**  $x \in \text{Rep-preal } r$  **and**  $\text{notin: } x + y \notin \text{Rep-preal } r$   
**using** *Dedekind-Real.Rep-preal Gleason9-34*  $\langle 0 < y \rangle$  **by** *blast*  
**then have**  $x + y \in \text{Rep-preal } (r + s)$  **using**  $y$   
**by** *(auto simp: mem-Rep-preal-add-iff)*  
**thus** *?thesis* **using**  $\text{notin}$  **by** *blast*  
**qed**

at last, Gleason prop. 9-3.5(iii) page 123

**proposition** *preal-self-less-add-left: (r::preal) < r + s*  
**by** *(meson Rep-preal-sum-not-subset not-less preal-le-def)*

## 1.10 Subtraction for Positive Reals

gleason prop. 9-3.5(iv), page 123: proving  $a < b \implies \exists d. a + d = b$ . We define the claimed  $D$  and show that it is a positive real

**lemma** *mem-diff-set:*

**assumes**  $r < s$

**shows**  $\text{cut } (\text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r))$

**proof** –

**obtain**  $p$  **where**  $\text{Rep-preal } r \subseteq \text{Rep-preal } s$   $p \in \text{Rep-preal } s$   $p \notin \text{Rep-preal } r$

**using** *assms* **unfolding** *preal-less-def* **by** *auto*

**then have**  $\{ \} \subset \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$

**apply** *(simp add: diff-set-def psubset-eq)*

**by** *(metis cut-Rep-preal add-eq-exists less-add-same-cancel1 preal-exists-greater preal-imp-pos)*

**moreover**

**obtain**  $q$  **where**  $q > 0$   $q \notin \text{Rep-preal } s$

**using** *Rep-preal-exists-bound* **by** *blast*

**then have**  $q \notin \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$

**by** *(auto simp: diff-set-def dest: cut-Rep-preal [THEN preal-downwards-closed])*

**moreover have**  $\text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r) \subset \{0 < ..\}$  **(is ?lhs < ?rhs)**

**using**  $\langle 0 < q \rangle$  *diff-set-def*  $q \notin$  **by** *blast*

**moreover have**  $z \in \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$

**if**  $u: u \in \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$  **and**  $0 < z < u$  **for**  $u z$

**using**  $u$  *that less-trans Rep-preal* **unfolding** *diff-set-def Dedekind-Real.cut-def*

**by** *auto*

**moreover have**  $\exists u \in \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r). y < u$

**if**  $y: y \in \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$  **for**  $y$

**proof** –

**obtain**  $a b$  **where**  $0 < a < b$   $a \notin \text{Rep-preal } r$   $a + y + b \in \text{Rep-preal } s$

**using**  $y$

**by** *(simp add: diff-set-def) (metis cut-Rep-preal add-eq-exists less-add-same-cancel1 preal-exists-greater)*

**then have**  $a + (y + b) \in \text{Rep-preal } s$

**by** *(simp add: add.assoc)*

**then have**  $y + b \in \text{diff-set } (\text{Rep-preal } s) (\text{Rep-preal } r)$

**using**  $\langle 0 < a \rangle \langle 0 < b \rangle \langle a \notin \text{Rep-preal } r \rangle y$

**by** *(auto simp: diff-set-def)*

**then show** *?thesis*

**using**  $\langle 0 < b \rangle$  *less-add-same-cancel1* **by** *blast*  
**qed**  
**ultimately show** *?thesis*  
**by** (*simp add: Dedekind-Real.cut-def*)  
**qed**

**lemma** *mem-Rep-preal-diff-iff*:

$r < s \implies$   
 $(z \in \text{Rep-preal } (s - r)) \longleftrightarrow$   
 $(\exists x. 0 < x \wedge 0 < z \wedge x \notin \text{Rep-preal } r \wedge x + z \in \text{Rep-preal } s)$   
**apply** (*simp add: preal-diff-def mem-diff-set Rep-preal*)  
**apply** (*force simp: diff-set-def*)  
**done**

**proposition** *less-add-left*:

**fixes**  $r::\text{preal}$   
**assumes**  $r < s$   
**shows**  $r + (s-r) = s$   
**proof** –  
**have**  $a + b \in \text{Rep-preal } s$   
**if**  $a \in \text{Rep-preal } r$   $c + b \in \text{Rep-preal } s$   $c \notin \text{Rep-preal } r$   
**and**  $0 < b$   $0 < c$  **for**  $a$   $b$   $c$   
**by** (*meson cut-Rep-preal add-less-imp-less-right add-pos-pos not-in-Rep-preal-ub preal-downwards-closed preal-imp-pos that*)  
**then have**  $r + (s-r) \leq s$   
**using** *assms mem-Rep-preal-add-iff mem-Rep-preal-diff-iff preal-le-def* **by** *auto*  
**have**  $x \in \text{Rep-preal } (r + (s - r))$  **if**  $x \in \text{Rep-preal } s$  **for**  $x$   
**proof** (*cases x \in Rep-preal r*)  
**case** *True*  
**then show** *?thesis*  
**using** *Rep-preal-self-subset* **by** *blast*  
**next**  
**case** *False*  
**have**  $\exists u v z. 0 < v \wedge 0 < z \wedge u \in \text{Rep-preal } r \wedge z \notin \text{Rep-preal } r \wedge z + v \in$   
 $\text{Rep-preal } s \wedge x = u + v$   
**if**  $x: x \in \text{Rep-preal } s$   
**proof** –  
**have**  $xpos: x > 0$   
**using** *Rep-preal preal-imp-pos that* **by** *blast*  
**obtain**  $e$  **where**  $epos: 0 < e$  **and**  $xe: x + e \in \text{Rep-preal } s$   
**by** (*metis cut-Rep-preal x add-eq-exists less-add-same-cancel1 preal-exists-greater*)  
**from** *Gleason9-34 [OF cut-Rep-preal epos]*  
**obtain**  $u$  **where**  $r: u \in \text{Rep-preal } r$  **and**  $notin: u + e \notin \text{Rep-preal } r$  ..  
**with**  $x$  *False xpos* **have**  $rless: u < x$  **by** (*blast intro: not-in-Rep-preal-ub*)  
**from** *add-eq-exists [of u x]*  
**obtain**  $y$  **where**  $eq: x = u + y$  **by** *auto*  
**show** *?thesis*  
**proof** (*intro exI conjI*)  
**show**  $u + e \notin \text{Rep-preal } r$  **by** (*rule notin*)

```

    show  $u + e + y \in \text{Rep-preal } s$  using  $xe \text{ eq}$  by (simp add: ac-simps)
    show  $0 < u + e$ 
      using epos preal-imp-pos [OF cut-Rep-preal r] by simp
    qed (use r rless eq in auto)
  qed
  then show ?thesis
    using assms mem-Rep-preal-add-iff mem-Rep-preal-diff-iff that by blast
  qed
  then have  $s \leq r + (s-r)$ 
    by (auto simp: preal-le-def)
  then show ?thesis
    by (simp add:  $\langle r + (s - r) \leq s \rangle$  antisym)
qed

```

```

lemma preal-add-less2-mono1:  $r < (s::\text{preal}) \implies r + t < s + t$ 
  by (metis add.assoc add.commute less-add-left preal-self-less-add-left)

```

```

lemma preal-add-less2-mono2:  $r < (s::\text{preal}) \implies t + r < t + s$ 
  by (auto intro: preal-add-less2-mono1 simp add: preal-add-commute [of t])

```

```

lemma preal-add-right-less-cancel:  $r + t < s + t \implies r < (s::\text{preal})$ 
  by (metis linorder-cases order.asym preal-add-less2-mono1)

```

```

lemma preal-add-left-less-cancel:  $t + r < t + s \implies r < (s::\text{preal})$ 
  by (auto elim: preal-add-right-less-cancel simp add: preal-add-commute [of t])

```

```

lemma preal-add-less-cancel-left [simp]:  $(t + (r::\text{preal}) < t + s) \longleftrightarrow (r < s)$ 
  by (blast intro: preal-add-less2-mono2 preal-add-left-less-cancel)

```

```

lemma preal-add-less-cancel-right [simp]:  $((r::\text{preal}) + t < s + t) = (r < s)$ 
  using preal-add-less-cancel-left [symmetric, of r s t] by (simp add: ac-simps)

```

```

lemma preal-add-le-cancel-left [simp]:  $(t + (r::\text{preal}) \leq t + s) = (r \leq s)$ 
  by (simp add: linorder-not-less [symmetric])

```

```

lemma preal-add-le-cancel-right [simp]:  $((r::\text{preal}) + t \leq s + t) = (r \leq s)$ 
  using preal-add-le-cancel-left [symmetric, of r s t] by (simp add: ac-simps)

```

```

lemma preal-add-right-cancel:  $(r::\text{preal}) + t = s + t \implies r = s$ 
  by (metis less-irrefl linorder-cases preal-add-less-cancel-right)

```

```

lemma preal-add-left-cancel:  $c + a = c + b \implies a = (b::\text{preal})$ 
  by (auto intro: preal-add-right-cancel simp add: preal-add-commute)

```

```

instance preal :: linordered-ab-semigroup-add

```

```

proof

```

```

  fix a b c :: preal

```

```

  show  $a \leq b \implies c + a \leq c + b$  by (simp only: preal-add-le-cancel-left)

```

```

qed

```

## 1.11 Completeness of type *preal*

Prove that supremum is a cut

Part 1 of Dedekind sections definition

**lemma** *preal-sup*:

**assumes** *le*:  $\bigwedge X. X \in P \implies X \leq Y$  **and**  $P \neq \{\}$

**shows** *cut*  $(\bigcup X \in P. \text{Rep-preal}(X))$

**proof** –

**have**  $\{\} \subset (\bigcup X \in P. \text{Rep-preal}(X))$

**using**  $\langle P \neq \{\} \rangle$  *mem-Rep-preal-Ex* **by** *fastforce*

**moreover**

**obtain** *q* **where**  $q > 0$  **and**  $q \notin (\bigcup X \in P. \text{Rep-preal}(X))$

**using** *Rep-preal-exists-bound* [of *Y*] *le* **by** (*auto simp: preal-le-def*)

**then have**  $(\bigcup X \in P. \text{Rep-preal}(X)) \subset \{0 < ..\}$

**using** *cut-Rep-preal preal-imp-pos* **by** *force*

**moreover**

**have**  $\bigwedge u z. \llbracket u \in (\bigcup X \in P. \text{Rep-preal}(X)); 0 < z; z < u \rrbracket \implies z \in (\bigcup X \in P. \text{Rep-preal}(X))$

**by** (*auto elim: cut-Rep-preal [THEN preal-downwards-closed]*)

**moreover**

**have**  $\bigwedge y. y \in (\bigcup X \in P. \text{Rep-preal}(X)) \implies \exists u \in (\bigcup X \in P. \text{Rep-preal}(X)). y < u$

**by** (*blast dest: cut-Rep-preal [THEN preal-exists-greater]*)

**ultimately show** *?thesis*

**by** (*simp add: Dedekind-Real.cut-def*)

**qed**

**lemma** *preal-psup-le*:

$\llbracket \bigwedge X. X \in P \implies X \leq Y; x \in P \rrbracket \implies x \leq \text{psup } P$

**using** *preal-sup* [of *P Y*] **unfolding** *preal-le-def psup-def* **by** *fastforce*

**lemma** *psup-le-ub*:  $\llbracket \bigwedge X. X \in P \implies X \leq Y; P \neq \{\} \rrbracket \implies \text{psup } P \leq Y$

**using** *preal-sup* [of *P Y*] **by** (*simp add: SUP-least preal-le-def psup-def*)

Supremum property

**proposition** *preal-complete*:

**assumes** *le*:  $\bigwedge X. X \in P \implies X \leq Y$  **and**  $P \neq \{\}$

**shows**  $(\exists X \in P. Z < X) \longleftrightarrow (Z < \text{psup } P)$  (**is** *?lhs = ?rhs*)

**proof**

**assume** *?lhs*

**then show** *?rhs*

**using** *preal-sup* [*OF assms*] *preal-less-def psup-def* **by** *auto*

**next**

**assume** *?rhs*

**then show** *?lhs*

**by** (*meson*  $\langle P \neq \{\} \rangle$  *not-less psup-le-ub*)

**qed**

## 1.12 Defining the Reals from the Positive Reals

Here we do quotients the old-fashioned way

**definition**

$realrel :: ((preal * preal) * (preal * preal)) \text{ set } \mathbf{where}$   
 $realrel = \{p. \exists x1\ y1\ x2\ y2. p = ((x1,y1),(x2,y2)) \wedge x1+y2 = x2+y1\}$

**definition**  $Real = UNIV // realrel$

**typedef**  $real = Real$

**morphisms**  $Rep-Real\ Abs-Real$

**unfolding**  $Real-def$  **by**  $(auto\ simp: quotient-def)$

This doesn't involve the overloaded "real" function: users don't see it

**definition**

$real-of-preal :: preal \Rightarrow real \mathbf{where}$   
 $real-of-preal\ m = Abs-Real\ (realrel\ \{\{(m + 1, 1)\}\})$

**instantiation**  $real :: \{zero, one, plus, minus, uminus, times, inverse, ord, abs, sgn\}$

**begin**

**definition**

$real-zero-def: 0 = Abs-Real(realrel\ \{\{(1, 1)\}\})$

**definition**

$real-one-def: 1 = Abs-Real(realrel\ \{\{(1 + 1, 1)\}\})$

**definition**

$real-add-def: z + w =$   
 $the\ elem\ (\bigcup (x,y) \in Rep-Real\ z. \bigcup (u,v) \in Rep-Real\ w.$   
 $\{ Abs-Real(realrel\ \{\{(x+u, y+v)\}\}) \})$

**definition**

$real-minus-def: - r = the\ elem\ (\bigcup (x,y) \in Rep-Real\ r. \{ Abs-Real(realrel\ \{\{(y,x)\}\})$   
 $\})$

**definition**

$real-diff-def: r - (s::real) = r + - s$

**definition**

$real-mult-def:$   
 $z * w =$   
 $the\ elem\ (\bigcup (x,y) \in Rep-Real\ z. \bigcup (u,v) \in Rep-Real\ w.$   
 $\{ Abs-Real(realrel\ \{\{(x*u + y*v, x*v + y*u)\}\}) \})$

**definition**

$real-inverse-def: inverse\ (r::real) \equiv (THE\ s. (r = 0 \wedge s = 0) \vee s * r = 1)$

**definition**

*real-divide-def*:  $r \text{ div } (s::\text{real}) \equiv r * \text{inverse } s$

**definition**

*real-le-def*:  $z \leq (w::\text{real}) \equiv$   
 $(\exists x y u v. x+v \leq u+y \wedge (x,y) \in \text{Rep-Real } z \wedge (u,v) \in \text{Rep-Real } w)$

**definition**

*real-less-def*:  $x < (y::\text{real}) \equiv x \leq y \wedge x \neq y$

**definition**

*real-abs-def*:  $|r::\text{real}| = (\text{if } r < 0 \text{ then } - r \text{ else } r)$

**definition**

*real-sgn-def*:  $\text{sgn } (x::\text{real}) = (\text{if } x=0 \text{ then } 0 \text{ else if } 0 < x \text{ then } 1 \text{ else } - 1)$

**instance ..**

**end**

### 1.13 Equivalence relation over positive reals

**lemma** *realrel-iff* [*simp*]:  $((x1,y1),(x2,y2)) \in \text{realrel} = (x1 + y2 = x2 + y1)$   
**by** (*simp add: realrel-def*)

**lemma** *preal-trans-lemma*:

**assumes**  $x + y1 = x1 + y$  **and**  $x + y2 = x2 + y$   
**shows**  $x1 + y2 = x2 + (y1::\text{preal})$   
**by** (*metis add.left-commute assms preal-add-left-cancel*)

**lemma** *equiv-realrel*: *equiv UNIV realrel*

**by** (*auto simp: equiv-def refl-on-def sym-def trans-def realrel-def intro: dest: preal-trans-lemma*)

Reduces equality of equivalence classes to the *Dedekind-Real.realrel* relation:  $(\text{Dedekind-Real.realrel } \{x\} = \text{Dedekind-Real.realrel } \{y\}) = ((x, y) \in \text{Dedekind-Real.realrel})$

**lemmas** *equiv-realrel-iff* [*simp*] =  
*eq-equiv-class-iff* [*OF equiv-realrel UNIV-I UNIV-I*]

**lemma** *realrel-in-real* [*simp*]:  $\text{realrel}\{x,y\} \in \text{Real}$   
**by** (*simp add: Real-def realrel-def quotient-def, blast*)

**declare** *Abs-Real-inject* [*simp*] *Abs-Real-inverse* [*simp*]

Case analysis on the representation of a real number as an equivalence class of pairs of positive reals.

**lemma** *eq-Abs-Real* [*case-names Abs-Real, cases type: real*]:  
 $(\bigwedge x y. z = \text{Abs-Real}(\text{realrel}\{x,y\}) \implies P) \implies P$

by (*metis Rep-Real-inverse prod.exhaust Rep-Real [of z, unfolded Real-def, THEN quotientE]*)

## 1.14 Addition and Subtraction

**lemma** *real-add*:

$$\begin{aligned} & \text{Abs-Real } (\text{realrel} \{x, y\}) + \text{Abs-Real } (\text{realrel} \{u, v\}) = \\ & \text{Abs-Real } (\text{realrel} \{x+u, y+v\}) \end{aligned}$$

**proof** –

have ( $\lambda z w. (\lambda(x,y). (\lambda(u,v). \{\text{Abs-Real } (\text{realrel} \{x+u, y+v\})\}) w) z$ )  
respects2 *realrel*

by (*clarsimp simp: congruent2-def*) (*metis add.left-commute preal-add-assoc*)

thus ?thesis

by (*simp add: real-add-def UN-UN-split-split-eq UN-equiv-class2 [OF equiv-realrel equiv-realrel]*)

qed

**lemma** *real-minus*:  $-\text{Abs-Real}(\text{realrel} \{x, y\}) = \text{Abs-Real}(\text{realrel} \{y, x\})$

**proof** –

have ( $\lambda(x,y). \{\text{Abs-Real } (\text{realrel} \{y, x\})\}$ ) respects *realrel*

by (*auto simp: congruent-def add.commute*)

thus ?thesis

by (*simp add: real-minus-def UN-equiv-class [OF equiv-realrel]*)

qed

**instance** *real* :: *ab-group-add*

**proof**

fix *x y z* :: *real*

show  $(x + y) + z = x + (y + z)$

by (*cases x, cases y, cases z, simp add: real-add add.assoc*)

show  $x + y = y + x$

by (*cases x, cases y, simp add: real-add add.commute*)

show  $0 + x = x$

by (*cases x, simp add: real-add real-zero-def ac-simps*)

show  $-x + x = 0$

by (*cases x, simp add: real-minus real-add real-zero-def add.commute*)

show  $x - y = x + -y$

by (*simp add: real-diff-def*)

qed

## 1.15 Multiplication

**lemma** *real-mult-congruent2-lemma*:

$$\begin{aligned} & \llbracket (x1 :: \text{preal}). \llbracket x1 + y2 = x2 + y1 \rrbracket \implies \\ & \quad x * x1 + y * y1 + (x * y2 + y * x2) = \\ & \quad x * x2 + y * y2 + (x * y1 + y * x1) \end{aligned}$$

by (*metis (no-types, opaque-lifting) add.left-commute preal-add-commute preal-add-mult-distrib2*)

**lemma** *real-mult-congruent2*:

$(\lambda p1 p2.$



```

      (λ(x1,y1). (λ(x2,y2).
        { Abs-Real (realrel“{(x1*x2 + y1*y2, x1*y2+y1*x2)} } ) p2) p1)
    respects2 realrel
  apply (rule congruent2-commuteI [OF equiv-realrel])
  by (auto simp: mult.commute add.commute combine-common-factor preal-add-assoc
    preal-add-commute)

```

```

lemma real-mult:
  Abs-Real((realrel“{(x1,y1)})) * Abs-Real((realrel“{(x2,y2)})) =
  Abs-Real(realrel “ {(x1*x2+y1*y2,x1*y2+y1*x2)} )
  by (simp add: real-mult-def UN-UN-split-split-eq
    UN-equiv-class2 [OF equiv-realrel equiv-realrel real-mult-congruent2])

```

```

lemma real-mult-commute: (z::real) * w = w * z
  by (cases z, cases w, simp add: real-mult ac-simps)

```

```

lemma real-mult-assoc: ((z1::real) * z2) * z3 = z1 * (z2 * z3)
  by (cases z1, cases z2, cases z3) (simp add: real-mult algebra-simps)

```

```

lemma real-mult-1: (1::real) * z = z
  by (cases z) (simp add: real-mult real-one-def algebra-simps)

```

```

lemma real-add-mult-distrib: ((z1::real) + z2) * w = (z1 * w) + (z2 * w)
  by (cases z1, cases z2, cases w) (simp add: real-add real-mult algebra-simps)

```

one and zero are distinct

```

lemma real-zero-not-eq-one: 0 ≠ (1::real)

```

```

proof –
  have (1::preal) < 1 + 1
    by (simp add: preal-self-less-add-left)
  then show ?thesis
    by (simp add: real-zero-def real-one-def neq-iff)
qed

```

```

instance real :: comm-ring-1

```

```

proof
  fix x y z :: real
  show (x * y) * z = x * (y * z) by (rule real-mult-assoc)
  show x * y = y * x by (rule real-mult-commute)
  show 1 * x = x by (rule real-mult-1)
  show (x + y) * z = x * z + y * z by (rule real-add-mult-distrib)
  show 0 ≠ (1::real) by (rule real-zero-not-eq-one)
qed

```

## 1.16 Inverse and Division

```

lemma real-zero-iff: Abs-Real (realrel “ {(x, x)} ) = 0
  by (simp add: real-zero-def add.commute)

```

```

lemma real-mult-inverse-left-ex:

```

```

assumes  $x \neq 0$  obtains  $y::real$  where  $y*x = 1$ 
proof (cases x)
  case (Abs-Real u v)
  show ?thesis
  proof (cases u v rule: linorder-cases)
    case less
    then have  $v * inverse (v - u) = 1 + u * inverse (v - u)$ 
      using less-add-left [of u v]
      by (metis preal-add-commute preal-add-mult-distrib preal-mult-inverse-right)
    then have Abs-Real (realrel“{(1, inverse (v-u) + 1)}”) *  $x - 1 = 0$ 
      by (simp add: Abs-Real real-mult preal-mult-inverse-right real-one-def) (simp
add: algebra-simps)
    with that show thesis by auto
  next
  case equal
  then show ?thesis
    using Abs-Real assms real-zero-iff by blast
  next
  case greater
  then have  $u * inverse (u - v) = 1 + v * inverse (u - v)$ 
    using less-add-left [of v u] by (metis add.commute distrib-right preal-mult-inverse-right)
  then have Abs-Real (realrel“{(inverse (u-v) + 1, 1)}”) *  $x - 1 = 0$ 
    by (simp add: Abs-Real real-mult preal-mult-inverse-right real-one-def) (simp
add: algebra-simps)
  with that show thesis by auto
qed
qed

```

**lemma** *real-mult-inverse-left*:

```

fixes  $x :: real$ 
assumes  $x \neq 0$  shows  $inverse x * x = 1$ 
proof -
  obtain  $y$  where  $y*x = 1$ 
    using assms real-mult-inverse-left-ex by blast
  then have (THE s.  $s * x = 1$ ) *  $x = 1$ 
  proof (rule theI)
    show  $y' = y$  if  $y' * x = 1$  for  $y'$ 
      by (metis < $y * x = 1$ > mult.left-commute mult.right-neutral that)
  qed
then show ?thesis
  using assms real-inverse-def by auto
qed

```

## 1.17 The Real Numbers form a Field

**instance**  $real :: field$

**proof**

**fix**  $x y z :: real$

```

show  $x \neq 0 \implies \text{inverse } x * x = 1$  by (rule real-mult-inverse-left)
show  $x / y = x * \text{inverse } y$  by (simp add: real-divide-def)
show  $\text{inverse } 0 = (0::\text{real})$  by (simp add: real-inverse-def)
qed

```

## 1.18 The $\leq$ Ordering

```

lemma real-le-refl:  $w \leq (w::\text{real})$ 
by (cases w, force simp: real-le-def)

```

The arithmetic decision procedure is not set up for type preal. This lemma is currently unused, but it could simplify the proofs of the following two lemmas.

```

lemma preal-eq-le-imp-le:
  assumes eq:  $a+b = c+d$  and le:  $c \leq a$ 
  shows  $b \leq (d::\text{preal})$ 
proof -
  from le have  $c+d \leq a+d$  by simp
  hence  $a+b \leq a+d$  by (simp add: eq)
  thus  $b \leq d$  by simp
qed

```

```

lemma real-le-lemma:
  assumes l:  $u1 + v2 \leq u2 + v1$ 
  and  $x1 + v1 = u1 + y1$ 
  and  $x2 + v2 = u2 + y2$ 
  shows  $x1 + y2 \leq x2 + (y1::\text{preal})$ 
proof -
  have  $(x1+v1) + (u2+y2) = (u1+y1) + (x2+v2)$  by (simp add: assms)
  hence  $(x1+y2) + (u2+v1) = (x2+y1) + (u1+v2)$  by (simp add: ac-simps)
  also have  $\dots \leq (x2+y1) + (u2+v1)$  by (simp add: assms)
  finally show ?thesis by simp
qed

```

```

lemma real-le:
   $\text{Abs-Real}(\text{realrel}\{\{x1,y1\}\}) \leq \text{Abs-Real}(\text{realrel}\{\{x2,y2\}\}) \iff x1 + y2 \leq x2 + y1$ 
  unfolding real-le-def by (auto intro: real-le-lemma)

```

```

lemma real-le-antisym:  $\llbracket z \leq w; w \leq z \rrbracket \implies z = (w::\text{real})$ 
by (cases z, cases w, simp add: real-le)

```

```

lemma real-trans-lemma:
  assumes  $x + v \leq u + y$ 
  and  $u + v' \leq u' + v$ 
  and  $x2 + v2 = u2 + y2$ 
  shows  $x + v' \leq u' + (y::\text{preal})$ 
proof -
  have  $(x+v') + (u+v) = (x+v) + (u+v')$  by (simp add: ac-simps)

```

```

also have ...  $\leq (u+y) + (u+v')$  by (simp add: assms)
also have ...  $\leq (u+y) + (u'+v)$  by (simp add: assms)
also have ...  $= (u'+y) + (u+v)$  by (simp add: ac-simps)
finally show ?thesis by simp
qed

lemma real-le-trans:  $\llbracket i \leq j; j \leq k \rrbracket \implies i \leq (k::real)$ 
by (cases i, cases j, cases k) (auto simp: real-le intro: real-trans-lemma)

instance real :: order
proof
show  $u < v \iff u \leq v \wedge \neg v \leq u$  for  $u\ v::real$ 
by (auto simp: real-less-def intro: real-le-antisym)
qed (auto intro: real-le-refl real-le-trans real-le-antisym)

instance real :: linorder
proof
show  $x \leq y \vee y \leq x$  for  $x\ y::real$ 
by (meson eq-refl le-cases real-le-def)
qed

instantiation real :: distrib-lattice
begin

definition
(inf :: real  $\Rightarrow$  real  $\Rightarrow$  real) = min

definition
(sup :: real  $\Rightarrow$  real  $\Rightarrow$  real) = max

instance
by standard (auto simp: inf-real-def sup-real-def max-min-distrib2)

end

```

## 1.19 The Reals Form an Ordered Field

```

lemma real-le-eq-diff:  $(x \leq y) \iff (x-y \leq (0::real))$ 
by (cases x, cases y) (simp add: real-le real-zero-def real-diff-def real-add real-minus
preal-add-commute)

lemma real-add-left-mono:
assumes  $le: x \leq y$  shows  $z + x \leq z + (y::real)$ 
proof -
have  $z + x - (z + y) = (z + -z) + (x - y)$ 
by (simp add: algebra-simps)
with  $le$  show ?thesis
by (simp add: real-le-eq-diff[of x] real-le-eq-diff[of z+x])
qed

```

**lemma** *real-sum-gt-zero-less*:  $(0 < s + (-w::real)) \implies (w < s)$   
**by** (*simp add: linorder-not-le [symmetric] real-le-eq-diff [of s]*)

**lemma** *real-less-sum-gt-zero*:  $(w < s) \implies (0 < s + (-w::real))$   
**by** (*simp add: linorder-not-le [symmetric] real-le-eq-diff [of s]*)

**lemma** *real-mult-order*:

**fixes**  $x\ y::real$   
**assumes**  $0 < x\ 0 < y$   
**shows**  $0 < x * y$   
**proof** (*cases x, cases y*)  
**show**  $0 < x * y$   
**if**  $x: x = Abs-Real (Dedekind-Real.realrel \{\{x1, x2\}\})$   
**and**  $y: y = Abs-Real (Dedekind-Real.realrel \{\{y1, y2\}\})$   
**for**  $x1\ x2\ y1\ y2$   
**proof** –  
**have**  $x2 < x1\ y2 < y1$   
**using** *assms not-le real-zero-def real-le x y*  
**by** (*metis preal-add-le-cancel-left real-zero-iff*) +  
**then obtain**  $xd\ yd$  **where**  $x1 = x2 + xd\ y1 = y2 + yd$   
**using** *less-add-left by metis*  
**then have**  $\neg (x * y \leq 0)$   
**apply** (*simp add: x y real-mult real-zero-def real-le*)  
**apply** (*simp add: not-le algebra-simps preal-self-less-add-left*)  
**done**  
**then show** *?thesis*  
**by** *auto*  
**qed**

**qed**

**lemma** *real-mult-less-mono2*:  $[(0::real) < z; x < y] \implies z * x < z * y$   
**by** (*metis add-uminus-conv-diff real-less-sum-gt-zero real-mult-order real-sum-gt-zero-less right-diff-distrib*)

**instance** *real :: linordered-field*

**proof**

**fix**  $x\ y\ z::real$   
**show**  $x \leq y \implies z + x \leq z + y$  **by** (*rule real-add-left-mono*)  
**show**  $|x| = (if\ x < 0\ then\ -x\ else\ x)$  **by** (*simp only: real-abs-def*)  
**show**  $sgn\ x = (if\ x=0\ then\ 0\ else\ if\ 0 < x\ then\ 1\ else\ -1)$   
**by** (*simp only: real-sgn-def*)  
**show**  $z * x < z * y$  **if**  $x < y\ 0 < z$   
**by** (*simp add: real-mult-less-mono2 that*)

**qed**

## 1.20 Completeness of the reals

The function *real-of-preal* requires many proofs, but it seems to be essential for proving completeness of the reals from that of the positive reals.

**lemma** *real-of-preal-add*:

*real-of-preal* ((*x*::preal) + *y*) = *real-of-preal* *x* + *real-of-preal* *y*  
**by** (*simp add: real-of-preal-def real-add algebra-simps*)

**lemma** *real-of-preal-mult*:

*real-of-preal* ((*x*::preal) \* *y*) = *real-of-preal* *x* \* *real-of-preal* *y*  
**by** (*simp add: real-of-preal-def real-mult algebra-simps*)

Gleason prop 9-4.4 p 127

**lemma** *real-of-preal-trichotomy*:

$\exists m. (x::real) = \text{real-of-preal } m \vee x = 0 \vee x = -(\text{real-of-preal } m)$

**proof** (*cases x*)

**case** (*Abs-Real u v*)

**show** *?thesis*

**proof** (*cases u v rule: linorder-cases*)

**case** *less*

**then show** *?thesis*

**using** *less-add-left*

**apply** (*simp add: Abs-Real real-of-preal-def real-minus real-zero-def*)

**by** (*metis preal-add-assoc preal-add-commute*)

**next**

**case** *equal*

**then show** *?thesis*

**using** *Abs-Real real-zero-iff* **by** *blast*

**next**

**case** *greater*

**then show** *?thesis*

**using** *less-add-left*

**apply** (*simp add: Abs-Real real-of-preal-def real-minus real-zero-def*)

**by** (*metis preal-add-assoc preal-add-commute*)

**qed**

**qed**

**lemma** *real-of-preal-less-iff* [*simp*]:

$(\text{real-of-preal } m1 < \text{real-of-preal } m2) = (m1 < m2)$

**by** (*metis not-less preal-add-less-cancel-right real-le real-of-preal-def*)

**lemma** *real-of-preal-le-iff* [*simp*]:

$(\text{real-of-preal } m1 \leq \text{real-of-preal } m2) = (m1 \leq m2)$

**by** (*simp add: linorder-not-less [symmetric]*)

**lemma** *real-of-preal-zero-less* [*simp*]:  $0 < \text{real-of-preal } m$

**by** (*metis less-add-same-cancel2 preal-self-less-add-left real-of-preal-add real-of-preal-less-iff*)

## 1.21 Theorems About the Ordering

**lemma** *real-gt-zero-preal-Ex*:  $(0 < x) \longleftrightarrow (\exists y. x = \text{real-of-preal } y)$   
**using** *order.asym real-of-preal-trichotomy* **by** *fastforce*

## 1.22 Completeness of Positive Reals

Supremum property for the set of positive reals

Let  $P$  be a non-empty set of positive reals, with an upper bound  $y$ . Then  $P$  has a least upper bound (written  $S$ ).

FIXME: Can the premise be weakened to  $\forall x \in P. x \leq y$ ?

**lemma** *posreal-complete*:

**assumes** *positive-P*:  $\forall x \in P. (0::\text{real}) < x$

**and** *not-empty-P*:  $\exists x. x \in P$

**and** *upper-bound-Ex*:  $\exists y. \forall x \in P. x < y$

**shows**  $\exists s. \forall y. (\exists x \in P. y < x) = (y < s)$

**proof** (*rule exI, rule allI*)

**fix**  $y$

**let**  $?pP = \{w. \text{real-of-preal } w \in P\}$

**show**  $(\exists x \in P. y < x) = (y < \text{real-of-preal } (\text{psup } ?pP))$

**proof** (*cases 0 < y*)

**assume** *neg-y*:  $\neg 0 < y$

**show** *?thesis*

**proof**

**assume**  $\exists x \in P. y < x$

**thus**  $y < \text{real-of-preal } (\text{psup } ?pP)$

**by** (*metis dual-order.strict-trans neg-y not-less-iff-gr-or-eq real-of-preal-zero-less*)

**next**

**assume**  $y < \text{real-of-preal } (\text{psup } ?pP)$

**obtain**  $x$  **where** *x-in-P*:  $x \in P$  **using** *not-empty-P ..*

**thus**  $\exists x \in P. y < x$  **using** *x-in-P*

**using** *neg-y not-less-iff-gr-or-eq positive-P* **by** *fastforce*

**qed**

**next**

**assume** *pos-y*:  $0 < y$

**then obtain**  $py$  **where** *y-is-py*:  $y = \text{real-of-preal } py$

**by** (*auto simp: real-gt-zero-preal-Ex*)

**obtain**  $a$  **where**  $a \in P$  **using** *not-empty-P ..*

**with** *positive-P* **have** *a-pos*:  $0 < a$  **..**

**then obtain**  $pa$  **where**  $a = \text{real-of-preal } pa$

**by** (*auto simp: real-gt-zero-preal-Ex*)

**hence**  $pa \in ?pP$  **using**  $\langle a \in P \rangle$  **by** *auto*

**hence** *pP-not-empty*:  $?pP \neq \{\}$  **by** *auto*

**obtain**  $sup$  **where** *sup*:  $\forall x \in P. x < sup$

**using** *upper-bound-Ex ..*

**from this and**  $\langle a \in P \rangle$  **have**  $a < \text{sup} ..$   
**hence**  $0 < \text{sup}$  **using**  $a\text{-pos}$  **by** *arith*  
**then obtain**  $\text{possup}$  **where**  $\text{sup} = \text{real-of-preal } \text{possup}$   
**by** (*auto simp: real-gt-zero-preal-Ex*)  
**hence**  $\forall X \in ?pP. X \leq \text{possup}$   
**using**  $\text{sup}$  **by** *auto*  
**with**  $pP\text{-not-empty}$  **have**  $\text{psup}: \bigwedge Z. (\exists X \in ?pP. Z < X) = (Z < \text{psup } ?pP)$   
**by** (*meson preal-complete*)  
**show** *?thesis*  
**proof**  
**assume**  $\exists x \in P. y < x$   
**then obtain**  $x$  **where**  $x\text{-in-}P: x \in P$  **and**  $y\text{-less-}x: y < x ..$   
**hence**  $0 < x$  **using**  $\text{pos-}y$  **by** *arith*  
**then obtain**  $px$  **where**  $x\text{-is-}px: x = \text{real-of-preal } px$   
**by** (*auto simp: real-gt-zero-preal-Ex*)  
  
**have**  $py\text{-less-}X: \exists X \in ?pP. py < X$   
**proof**  
**show**  $py < px$  **using**  $y\text{-is-}py$  **and**  $x\text{-is-}px$  **and**  $y\text{-less-}x$   
**by** *simp*  
**show**  $px \in ?pP$  **using**  $x\text{-in-}P$  **and**  $x\text{-is-}px$  **by** *simp*  
**qed**  
  
**have**  $(\exists X \in ?pP. py < X) \implies (py < \text{psup } ?pP)$   
**using**  $\text{psup}$  **by** *simp*  
**hence**  $py < \text{psup } ?pP$  **using**  $py\text{-less-}X$  **by** *simp*  
**thus**  $y < \text{real-of-preal } (\text{psup } \{w. \text{real-of-preal } w \in P\})$   
**using**  $y\text{-is-}py$  **and**  $\text{pos-}y$  **by** *simp*  
**next**  
**assume**  $y\text{-less-}psup: y < \text{real-of-preal } (\text{psup } ?pP)$   
  
**hence**  $py < \text{psup } ?pP$  **using**  $y\text{-is-}py$   
**by** *simp*  
**then obtain**  $X$  **where**  $py\text{-less-}X: py < X$  **and**  $X\text{-in-}pP: X \in ?pP$   
**using**  $\text{psup}$  **by** *auto*  
**then obtain**  $x$  **where**  $x\text{-is-}X: x = \text{real-of-preal } X$   
**by** (*simp add: real-gt-zero-preal-Ex*)  
**hence**  $y < x$  **using**  $py\text{-less-}X$  **and**  $y\text{-is-}py$   
**by** *simp*  
**moreover have**  $x \in P$   
**using**  $x\text{-is-}X$  **and**  $X\text{-in-}pP$  **by** *simp*  
**ultimately show**  $\exists x \in P. y < x ..$   
**qed**  
**qed**  
**qed**

### 1.23 Completeness

lemma *reals-complete*:



**fixes**  $S :: \text{real set}$   
**assumes**  $\text{notempty-}S: \exists X. X \in S$   
**and**  $\text{exists-Ub}: \text{bdd-above } S$   
**shows**  $\exists x. (\forall s \in S. s \leq x) \wedge (\forall y. (\forall s \in S. s \leq y) \longrightarrow x \leq y)$   
**proof** –  
**obtain**  $X$  **where**  $X\text{-in-}S: X \in S$  **using**  $\text{notempty-}S$  ..  
**obtain**  $Y$  **where**  $Y\text{-isUb}: \forall s \in S. s \leq Y$   
**using**  $\text{exists-Ub}$  **by** ( $\text{auto simp: bdd-above-def}$ )  
**let**  $?SHIFT = \{z. \exists x \in S. z = x + (-X) + 1\} \cap \{x. 0 < x\}$   
  
{  
  **fix**  $x$   
  **assume**  $S\text{-le-}x: \forall s \in S. s \leq x$   
  {  
    **fix**  $s$   
    **assume**  $s \in \{z. \exists x \in S. z = x + -X + 1\}$   
    **hence**  $\exists x \in S. s = x + -X + 1$  ..  
    **then obtain**  $x1$  **where**  $x1: x1 \in S \ s = x1 + (-X) + 1$  ..  
    **then have**  $x1 \leq x$  **using**  $S\text{-le-}x$  **by**  $\text{simp}$   
    **with**  $x1$  **have**  $s \leq x + -X + 1$  **by**  $\text{arith}$   
  }  
  **then have**  $\forall s \in ?SHIFT. s \leq x + (-X) + 1$   
  **by**  $\text{auto}$   
} **note**  $S\text{-Ub-is-SHIFT-Ub} = \text{this}$   
  
**have**  $*$ :  $\forall s \in ?SHIFT. s \leq Y + (-X) + 1$  **using**  $Y\text{-isUb}$  **by** ( $\text{rule } S\text{-Ub-is-SHIFT-Ub}$ )  
**have**  $\forall s \in ?SHIFT. s < Y + (-X) + 2$   
**proof**  
  **fix**  $s$  **assume**  $s \in ?SHIFT$   
  **with**  $*$  **have**  $s \leq Y + (-X) + 1$  **by**  $\text{simp}$   
  **also have**  $\dots < Y + (-X) + 2$  **by**  $\text{simp}$   
  **finally show**  $s < Y + (-X) + 2$  .  
**qed**  
**moreover have**  $\forall y \in ?SHIFT. 0 < y$  **by**  $\text{auto}$   
**moreover have**  $\text{shifted-not-empty}: \exists u. u \in ?SHIFT$   
  **using**  $X\text{-in-}S$  **and**  $Y\text{-isUb}$  **by**  $\text{auto}$   
**ultimately obtain**  $t$  **where**  $t\text{-is-Lub}: \forall y. (\exists x \in ?SHIFT. y < x) = (y < t)$   
  **using**  $\text{posreal-complete [of ?SHIFT]}$  **unfolding**  $\text{bdd-above-def}$  **by**  $\text{blast}$   
  
**show**  $?thesis$   
**proof**  
  **show**  $(\forall s \in S. s \leq (t + X + (-1))) \wedge (\forall y. (\forall s \in S. s \leq y) \longrightarrow (t + X + (-1))$   
 $\leq y)$   
  **proof**  $\text{safe}$   
  **fix**  $x$   
  **assume**  $\forall s \in S. s \leq x$   
  **hence**  $\forall s \in ?SHIFT. s \leq x + (-X) + 1$   
  **using**  $S\text{-Ub-is-SHIFT-Ub}$  **by**  $\text{simp}$   
  **then have**  $\neg x + (-X) + 1 < t$

```

    by (subst t-is-Lub[rule-format, symmetric]) (simp add: not-less)
  thus  $t + X + -1 \leq x$  by arith
next
fix y
assume y-in-S:  $y \in S$ 
obtain u where u-in-shift:  $u \in ?SHIFT$  using shifted-not-empty ..
hence  $\exists x \in S. u = x + -X + 1$  by simp
then obtain x where x-and-u:  $u = x + -X + 1$  ..
have u-le-t:  $u \leq t$ 
proof (rule dense-le)
  fix x assume  $x < u$  then have  $x < t$ 
    using u-in-shift t-is-Lub by auto
  then show  $x \leq t$  by simp
qed

show  $y \leq t + X + -1$ 
proof cases
  assume  $y \leq x$ 
  moreover have  $x = u + X + -1$  using x-and-u by arith
  moreover have  $u + X + -1 \leq t + X + -1$  using u-le-t by arith
  ultimately show  $y \leq t + X + -1$  by arith
next
assume  $\sim(y \leq x)$ 
hence x-less-y:  $x < y$  by arith

have  $x + (-X) + 1 \in ?SHIFT$  using x-and-u and u-in-shift by simp
hence  $0 < x + (-X) + 1$  by simp
hence  $0 < y + (-X) + 1$  using x-less-y by arith
hence *:  $y + (-X) + 1 \in ?SHIFT$  using y-in-S by simp
have  $y + (-X) + 1 \leq t$ 
proof (rule dense-le)
  fix x assume  $x < y + (-X) + 1$  then have  $x < t$ 
    using * t-is-Lub by auto
  then show  $x \leq t$  by simp
qed
thus ?thesis by simp
qed
qed
qed
qed

```

## 1.24 The Archimedean Property of the Reals

**theorem** *reals-Archimedean*:

fixes  $x :: real$

assumes  $x\text{-pos}: 0 < x$

shows  $\exists n. \text{inverse } (\text{of-nat } (\text{Suc } n)) < x$

**proof** (rule ccontr)

assume *contr*:  $\neg ?thesis$

```

have  $\forall n. x * \text{of-nat} (\text{Suc } n) \leq 1$ 
proof
  fix  $n$ 
  from contr have  $x \leq \text{inverse} (\text{of-nat} (\text{Suc } n))$ 
    by (simp add: linorder-not-less)
  hence  $x \leq (1 / (\text{of-nat} (\text{Suc } n)))$ 
    by (simp add: inverse-eq-divide)
  moreover have  $(0::\text{real}) \leq \text{of-nat} (\text{Suc } n)$ 
    by (rule of-nat-0-le-iff)
  ultimately have  $x * \text{of-nat} (\text{Suc } n) \leq (1 / \text{of-nat} (\text{Suc } n)) * \text{of-nat} (\text{Suc } n)$ 
    by (rule mult-right-mono)
  thus  $x * \text{of-nat} (\text{Suc } n) \leq 1$  by (simp del: of-nat-Suc)
qed
hence  $2: \text{bdd-above} \{z. \exists n. z = x * (\text{of-nat} (\text{Suc } n))\}$ 
  by (auto intro!: bdd-aboveI[of - 1])
have  $1: \exists X. X \in \{z. \exists n. z = x * (\text{of-nat} (\text{Suc } n))\}$  by auto
obtain  $t$  where
  upper:  $\bigwedge z. z \in \{z. \exists n. z = x * \text{of-nat} (\text{Suc } n)\} \implies z \leq t$  and
  least:  $\bigwedge y. (\bigwedge a. a \in \{z. \exists n. z = x * \text{of-nat} (\text{Suc } n)\} \implies a \leq y) \implies t \leq y$ 
  using reals-complete[OF 1 2] by auto

have  $t \leq t + - x$ 
proof (rule least)
  fix  $a$  assume  $a: a \in \{z. \exists n. z = x * (\text{of-nat} (\text{Suc } n))\}$ 
  have  $\forall n::\text{nat}. x * \text{of-nat } n \leq t + - x$ 
  proof
    fix  $n$ 
    have  $x * \text{of-nat} (\text{Suc } n) \leq t$ 
      by (simp add: upper)
    hence  $x * (\text{of-nat } n) + x \leq t$ 
      by (simp add: distrib-left)
    thus  $x * (\text{of-nat } n) \leq t + - x$  by arith
  qed hence  $\forall m. x * \text{of-nat} (\text{Suc } m) \leq t + - x$  by (simp del: of-nat-Suc)
  with  $a$  show  $a \leq t + - x$ 
    by auto
  qed
thus False using x-pos by arith
qed

```

There must be other proofs, e.g. *Suc* of the largest integer in the cut representing  $x$ .

**lemma** *reals-Archimedean2*:  $\exists n. (x::\text{real}) < \text{of-nat} (n::\text{nat})$

**proof** *cases*

**assume**  $x \leq 0$

**hence**  $x < \text{of-nat} (1::\text{nat})$  **by** *simp*

**thus** *?thesis ..*

**next**

**assume**  $\neg x \leq 0$

**hence** *x-greater-zero*:  $0 < x$  **by** *simp*

```

hence  $0 < \text{inverse } x$  by simp
then obtain  $n$  where  $\text{inverse } (\text{of-nat } (\text{Suc } n)) < \text{inverse } x$ 
  using reals-Archimedean by blast
hence  $\text{inverse } (\text{of-nat } (\text{Suc } n)) * x < \text{inverse } x * x$ 
  using x-greater-zero by (rule mult-strict-right-mono)
hence  $\text{inverse } (\text{of-nat } (\text{Suc } n)) * x < 1$ 
  using x-greater-zero by simp
hence  $\text{of-nat } (\text{Suc } n) * (\text{inverse } (\text{of-nat } (\text{Suc } n)) * x) < \text{of-nat } (\text{Suc } n) * 1$ 
  by (rule mult-strict-left-mono) (simp del: of-nat-Suc)
hence  $x < \text{of-nat } (\text{Suc } n)$ 
  by (simp add: algebra-simps del: of-nat-Suc)
thus  $\exists (n::\text{nat}). x < \text{of-nat } n$  ..
qed

```

```

instance real :: archimedean-field
proof
  fix  $r :: \text{real}$ 
  obtain  $n :: \text{nat}$  where  $r < \text{of-nat } n$ 
    using reals-Archimedean2 ..
  then have  $r \leq \text{of-int } (\text{int } n)$ 
    by simp
  then show  $\exists z. r \leq \text{of-int } z$  ..
qed

```

**end**

## References

- [1] J. D. Fleuriot. On the mechanization of real analysis in Isabelle/HOL. In M. Aagaard and J. Harrison, editors, *Theorem Proving in Higher Order Logics*, volume LNCS 1869, pages 145–161. Springer, 2000.
- [2] J. D. Fleuriot and L. C. Paulson. Mechanizing nonstandard real analysis. *LMS Journal of Computation and Mathematics*, 3:140–190, 2000. <http://www.lms.ac.uk/jcm/3/lms1999-027/>.
- [3] A. Gleason. *Fundamentals of Abstract Analysis*. Taylor & Francis, 1991.
- [4] L. Jutting. *Checking Landau’s “Grundlagen” in the AUTOMATH System*. PhD thesis, Eindhoven University of Technology, 1977. <https://doi.org/10.6100/IR23183>.