

Eudoxus Reals

Ata Keskin

May 26, 2024

Abstract

In this project, we present a peculiar construction of the real numbers, called “Eudoxus reals”, using Isabelle/HOL. Similar to the classical method of Dedekind cuts, our approach starts from first principles. However, unlike Dedekind cuts, Eudoxus reals directly derive real numbers from integers, bypassing the intermediate step of constructing rational numbers.

This construction of the real numbers was first discovered by Stephen Schanuel. Schanuel named his construction after the ancient Greek philosopher Eudoxus, who developed a theory of magnitude and proportion to explain the relations between the discrete and the continuous. Our formalization is based on R.D. Arthan’s paper detailing the construction [1]. For establishing the existence of multiplicative inverses for positive slopes, we used the idea of finding a suitable representative from Sławomir Kołodyński’s construction on IsarMathLib which is based on Zermelo–Fraenkel set theory.

Contents

1	Slopes	2
1.1	Bounded Functions	2
1.2	Properties of Slopes	3
1.3	Set Membership of <i>Inf</i> and <i>Sup</i> on Integers	11
2	Eudoxus Reals	13
2.1	Type Definition	13
2.2	Addition and Subtraction	15
2.3	Multiplication	17
2.4	Ordering	20
2.5	Multiplicative Inverse	30
2.6	Completeness	37

```

theory Slope
imports HOL.Archimedean-Field
begin

```

1 Slopes

1.1 Bounded Functions

```

definition bounded :: ('a ⇒ int) ⇒ bool where
  bounded f ←→ bdd-above ((λz. |f z|) ‘ UNIV)

```

```

lemma boundedI:
  assumes ∧z. |f z| ≤ C
  shows bounded f
  unfolding bounded-def by (rule bdd-aboveI2, force intro: assms)

```

```

lemma boundedE[elim]:
  assumes bounded f ∃ C. (∀z. |f z| ≤ C) ∧ 0 ≤ C ⇒ P
  shows P
  using assms unfolding bounded-def bdd-above-def by fastforce

```

```

lemma boundedE-strict:
  assumes bounded f ∃ C. (∀z. |f z| < C) ∧ 0 < C ⇒ P
  shows P
  by (meson bounded-def bdd-above-def assms boundedE gt-ex order.strict-trans1)

```

```

lemma bounded-alt-def: bounded f ←→ (∃ C. ∀z. |f z| ≤ C) using boundedI boundedE by meson

```

```

lemma bounded-iff-finite-range: bounded f ←→ finite (range f)
proof
  assume bounded f
  then obtain C where bound: |z| ≤ C if z ∈ range f for z by blast
  have range f ⊆ {z. z ≤ C ∧ -z ≤ C} using abs-le-D1[OF bound] abs-le-D2[OF bound] by blast
  also have ... = {(-C)..C} by force
  finally show finite (range f) using finite-subset finite-atLeastAtMost-int by blast
next
  assume finite (range f)
  hence |f z| ≤ max (abs (Sup (range f))) (abs (Inf (range f))) for z
  using cInf-lower[OF - bdd-below-finite, of f z range f] cSup-upper[OF - bdd-above-finite, of f z range f] by force
  thus bounded f by (rule boundedI)
qed

```

```

lemma bounded-constant:
  shows bounded (λ-. c)

```

by (rule boundedI[of - |c|], blast)

lemma *bounded-add*:

assumes *bounded f bounded g*

shows *bounded (λz. f z + g z)*

proof –

obtain *C-f C-g* where $|f z| \leq C-f$ $|g z| \leq C-g$ for *z* using *assms* by *blast*

hence $|f z + g z| \leq C-f + C-g$ for *z* by (*meson abs-triangle-ineq add-mono dual-order.trans*)

thus *?thesis* by (*blast intro: boundedI*)

qed

lemma *bounded-mult*:

assumes *bounded f bounded g*

shows *bounded (λz. f z * g z)*

proof –

obtain *C* where *bound*: $|f z| \leq C$ and *C-nonneg*: $0 \leq C$ for *z* using *assms* by *blast*

obtain *C'* where *bound'*: $|g z| \leq C'$ for *z* using *assms* by *blast*

show *?thesis* using *mult-mono[OF bound bound' C-nonneg abs-ge-zero]* by (*simp only: boundedI[of λz. f z * g z C * C'] abs-mult*)

qed

lemma *bounded-mult-const*:

assumes *bounded f*

shows *bounded (λz. c * f z)*

by (rule *bounded-mult[OF bounded-constant[of c] assms]*)

lemma *bounded-uminus*:

assumes *bounded f*

shows *bounded (λx. - f x)*

using *bounded-mult-const[OF assms, of - 1]* by *simp*

lemma *bounded-comp*:

assumes *bounded f*

shows *bounded (f o g)* and *bounded (g o f)*

proof –

show *bounded (f o g)* using *assms boundedI comp-def boundedE* by *metis*

next

have *range (g o f) = g ' range f* by *fastforce*

thus *bounded (g o f)* using *assms* by (*fastforce simp: bounded-iff-finite-range*)

qed

1.2 Properties of Slopes

definition *slope* :: $(int \Rightarrow int) \Rightarrow bool$ where

slope f \iff *bounded (λ(m, n). f (m + n) - (f m + f n))*

lemma *bounded-slopeI*:

assumes *bounded f*
shows *slope f*
proof –
obtain C **where** $|f x| \leq C$ **for** x **using** *assms* **by** *blast*
hence $|f (m + n) - (f m + f n)| \leq C + (C + C)$ **for** $m n$
using *abs-triangle-ineq4*[*of f (m + n) f m + f n*] *abs-triangle-ineq*[*of f m f n*]
by (*meson add-mono order-trans*)
thus *?thesis* **unfolding** *slope-def* **by** (*fast intro: boundedI*)
qed

lemma *slopeE[elim]*:
assumes *slope f*
obtains C **where** $\bigwedge m n. |f (m + n) - (f m + f n)| \leq C$ $0 \leq C$ **using** *assms*
unfolding *slope-def* **by** *fastforce*

lemma *slope-add*:
assumes *slope f slope g*
shows *slope ($\lambda z. f z + g z$)*
proof –
obtain C **where** *bound*: $|f (m + n) - (f m + f n)| \leq C$ **for** $m n$ **using** *assms*
by *fast*
obtain C' **where** *bound'*: $|g (m + n) - (g m + g n)| \leq C'$ **for** $m n$ **using** *assms*
by *fast*
have $|f (m + n) - (f m + f n)| + |g (m + n) - (g m + g n)| \leq C + C'$ **for** m
 n **using** *add-mono-thms-linordered-semiring(1)* *bound bound'* **by** *fast*
moreover **have** $|(\lambda z. f z + g z) (m + n) - ((\lambda z. f z + g z) m + (\lambda z. f z + g z) n)| \leq |f (m + n) - (f m + f n)| + |g (m + n) - (g m + g n)|$ **for** $m n$ **by**
linarith
ultimately **have** $|(\lambda z. f z + g z) (m + n) - ((\lambda z. f z + g z) m + (\lambda z. f z + g z) n)| \leq C + C'$ **for** $m n$ **using** *order-trans* **by** *fast*
thus *slope ($\lambda z. f z + g z$)* **unfolding** *slope-def* **by** (*fast intro: boundedI*)
qed

lemma *slope-symmetric-bound*:
assumes *slope f*
obtains C **where** $\bigwedge p q. |p * f q - q * f p| \leq (|p| + |q| + 2) * C$ $0 \leq C$
proof –
obtain C **where** *bound*: $|f (m + n) - (f m + f n)| \leq C$ **and** *C-nonneg*: $0 \leq C$
for $m n$ **using** *assms* **by** *fast*

have $*$: $|f (p * q) - p * f q| \leq (|p| + 1) * C$ **for** $p q$
proof (*induction p rule: int-induct[where ?k=0]*)
case *base*
then **show** *?case* **using** *bound[of 0 0]* **by** *force*
next
case (*step1 p*)
have $|f ((p + 1) * q) - f (p * q) - f q| \leq C$ **using** *bound[of p * q q]* **by**
(*auto simp: distrib-left mult.commute*)
hence $|f ((p + 1) * q) - f q - p * f q| \leq C + (|p| + 1) * C$ **using** *step1* **by**

fastforce
thus *?case using step1 by (auto simp add: distrib-left mult.commute)*
next
case (*step2 p*)
have $|f ((p - 1) * q) + f q - f (p * q)| \leq C$ **using** *bound[of p * q - q q] by (auto simp: mult.commute right-diff-distrib[^])*
hence $|f ((p - 1) * q) + f q - p * f q| \leq C + (|p| + 1) * C$ **using** *step2 by force*
hence $|f ((p - 1) * q) - (p - 1) * f q| \leq C + (|p - 1|) * C$ **using** *step2 by (auto simp: mult.commute right-diff-distrib[^])*
thus *?case by (auto simp add: distrib-left mult.commute)*
qed

have $|p * f q - q * f p| \leq (|p| + |q| + 2) * C$ **for** *p q*
proof -
have $|p * f q - q * f p| \leq |f (p * q) - p * f q| + |f (q * p) - q * f p|$ **by** *(fastforce simp: mult.commute)*
also have $\dots \leq (|p| + 1) * C + (|q| + 1) * C$ **using** **[of p q] *[of q p] by fastforce*
also have $\dots = (|p| + |q| + 2) * C$ **by algebra**
finally show *?thesis .*
qed
thus *?thesis using that C-nonneg by blast*
qed

lemma *slope-linear-bound:*
assumes *slope f*
obtains *A B where $\forall n. |f n| \leq A * |n| + B$ $0 \leq A$ $0 \leq B$*
proof -
obtain *C where bound: $|p * f q - q * f p| \leq (|p| + |q| + 2) * C$ $0 \leq C$ for p q* **using** *assms slope-symmetric-bound by blast*

have $|f p| \leq (C + |f 1|) * |p| + 3 * C$ **for** *p*
proof -
have $|p * f 1 - f p| \leq (|p| + 3) * C$ **using** *bound(1)[of - 1] by (simp add: add.commute)*
hence $|f p - p * f 1| \leq (|p| + 3) * C$ **by** *(subst abs-minus[of f p - p * f 1, symmetric], simp)*
hence $|f p| \leq (|p| + 3) * C + |p * f 1|$ **using** *dual-order.trans abs-triangle-ineq2 diff-le-eq by fast*
hence $|f p| \leq |p| * C + 3 * C + |p| * |f 1|$ **by** *(simp add: abs-mult int-distrib(2) mult.commute)*
hence $|f p| \leq |p| * (C + |f 1|) + 3 * C$ **by** *(simp add: ring-class.ring-distrib(1))*
thus *?thesis using mult.commute by metis*
qed
thus *?thesis using that bound(2) by fastforce*
qed

lemma *slope-comp:*

assumes *slope f slope g*
shows *slope (f o g)*
proof –
obtain *C* **where** *bound: |f (m + n) - (f m + f n)| ≤ C* **for** *m n* **using** *assms*
by *fast*
obtain *C'* **where** *bound': |g (m + n) - (g m + g n)| ≤ C'* **for** *m n* **using** *assms*
by *fast*
obtain *A B* **where** *f-linear-bound: |f n| ≤ A * |n| + B 0 ≤ A 0 ≤ B* **for** *n*
using *slope-linear-bound[OF assms(1)]* **by** *blast*
{
 fix *m n*
 have $|f (g (m + n)) - (f (g m) + f (g n))| ≤ (|f (g (m + n)) - f (g m + g n)| + |f (g m + g n) - (f (g m) + f (g n))| :: int)$ **by** *linarith*
 also have $... ≤ |f (g (m + n)) - f (g m + g n)| + C$ **using** *bound[of g m g n]* **by** *auto*
 also have $... ≤ |f (g (m + n)) - f (g m + g n) - f (g (m + n) - (g m + g n))| + |f (g (m + n) - (g m + g n))| + C$ **by** *fastforce*
 also have $... ≤ |f (g (m + n) - (g m + g n))| + 2 * C$ **using** *bound[of g (m + n) - (g m + g n) (g m + g n)]* **by** *fastforce*
 also have $... ≤ A * |g (m + n) - (g m + g n)| + B + 2 * C$ **using** *f-linear-bound(1)[of g (m + n) - (g m + g n)]* **by** *linarith*
 also have $... ≤ A * C' + B + 2 * C$ **using** *mult-left-mono[OF bound'[of m n], OF f-linear-bound(2)]* **by** *presburger*
 finally have $|f (g (m + n)) - (f (g m) + f (g n))| ≤ A * C' + B + 2 * C$
by *blast*
}
thus *slope (f o g) unfolding comp-def slope-def by (fast intro: boundedI)*
qed

lemma *slope-scale: slope ((* a) by (auto simp add: slope-def distrib-left intro: boundedI)*

lemma *slope-zero: slope (λ-. 0) using slope-scale[of 0] by (simp add: lambda-zero)*

lemma *slope-one: slope id using slope-scale[of 1] by (simp add: slope-def)*

lemma *slope-uminus: slope uminus using slope-scale[of -1] by (simp add: slope-def)*

lemma *slope-uminus':*

assumes *slope f*

shows *slope (λx. - f x)*

using *slope-comp[OF slope-uminus assms]* **by** *(simp add: slope-def)*

lemma *slope-minus:*

assumes *slope f slope g*

shows *slope (λx. f x - g x)*

using *slope-add[OF assms(1) slope-uminus', OF assms(2)]* **by** *simp*

lemma *slope-comp-commute:*

assumes *slope f slope g*
shows *bounded* ($\lambda z. (f \circ g) z - (g \circ f) z$)
proof –
obtain C **where** *bound*: $|z * f (g z) - (g z) * (f z)| \leq (|z| + |g z| + 2) * C \ 0 \leq C$ **for** z **using** *slope-symmetric-bound*[*OF assms(1)*] **by** *metis*
obtain C' **where** *bound'*: $|(f z) * (g z) - z * g (f z)| \leq (|f z| + |z| + 2) * C' \ 0 \leq C'$ **for** z **using** *slope-symmetric-bound*[*OF assms(2)*] **by** *metis*

obtain $A \ B$ **where** *f-lbound*: $|f z| \leq A * |z| + B \ 0 \leq A \ 0 \leq B$ **for** z **using** *slope-linear-bound*[*OF assms(1)*] **by** *blast*
obtain $A' \ B'$ **where** *g-lbound*: $|g z| \leq A' * |z| + B' \ 0 \leq A' \ 0 \leq B'$ **for** z **using** *slope-linear-bound*[*OF assms(2)*] **by** *blast*

have *combined-bound*: $|z * f (g z) - z * g (f z)| \leq (|z| + |g z| + 2) * C + (|f z| + |z| + 2) * C'$ **for** z
by (*intro order-trans*[*OF - add-mono*[*OF bound(1) bound'(1)*]]) (*fastforce simp add: mult.commute*[*of f z g z*])

{
 fix z
 define $D \ E$ **where** $D = (C + C' + A' * C + A * C')$ **and** $E = (2 + B') * C + (2 + B) * C'$
 have *E-nonneg*: $0 \leq E$ **unfolding** *E-def* **using** *g-lbound bound f-lbound bound'* **by** *simp*
 have *D-nonneg*: $0 \leq D$ **unfolding** *D-def* **using** *g-lbound bound f-lbound bound'* **by** *simp*

 have $(|z| + |g z| + 2) * C + (|f z| + |z| + 2) * C' = |z| * (C + C') + |g z| * C + |f z| * C' + 2 * C + 2 * C'$ **by** *algebra*
 hence $|z| * |f (g z) - g (f z)| \leq |z| * (C + C') + |g z| * C + |f z| * C' + 2 * C + 2 * C'$ **using** *combined-bound right-diff-distrib abs-mult* **by** *metis*
 also have $\dots \leq |z| * (C + C') + (A' * |z| + B') * C + |f z| * C' + 2 * C + 2 * C'$ **using** *mult-right-mono*[*OF g-lbound(1)*][*of z*] *bound(2)*] **by** *presburger*
 also have $\dots \leq |z| * (C + C') + (A' * |z| + B') * C + (A * |z| + B) * C' + 2 * C + 2 * C'$ **using** *mult-right-mono*[*OF f-lbound(1)*][*of z*] *bound'(2)*] **by** *presburger*
 also have $\dots = |z| * (C + C' + A' * C + A * C') + (2 + B') * C + (2 + B) * C'$ **by** *algebra*
 finally have $|z| * |f (g z) - g (f z)| \leq |z| * D + E$ **unfolding** *D-def E-def* **by** *presburger*
 have $|f (g z) - g (f z)| \leq D + E + |f (g 0) - g (f 0)|$
 proof (*cases z = 0*)
 case *True*
 then show *?thesis* **using** *D-nonneg E-nonneg* **by** *fastforce*
 next
 case *False*
 have $|z| * |f (g z) - g (f z)| \leq |z| * (D + E)$
 using *mult-right-mono*[*OF Ints-nonzero-abs-ge1*][*OF - False*] *E-nonneg*] *distrib-left*[*of |z| D E*] *

by (*simp add: Ints-def*)
 hence $|f (g z) - g (f z)| \leq D + E$ using *False by simp*
 thus *?thesis by linarith*
 qed
 }
 thus *?thesis by (fastforce intro: boundedI)*
 qed

lemma *int-set-infiniteI*:
 assumes $\bigwedge C. C \geq 0 \implies \exists N \geq C. N \in (A :: \text{int set})$
 shows *infinite A*
 by (*meson assms abs-ge-zero abs-le-iff gt-ex le-cSup-finite linorder-not-less order-less-le-trans*)

lemma *int-set-infiniteD*:
 assumes *infinite (A :: int set) C ≥ 0*
 obtains *z where $z \in A$ $C \leq |z|$*
proof –
 {
 assume *asm: $\forall z \in A. C > |z|$*
 let *?f = $\lambda z. (if z \in A then z else (0::int))$*
 have *bounded: $\forall z \in A. |?f z| \leq C$ using *asm by fastforce**
 moreover have $\forall z \in UNIV - A. |?f z| \leq C$ using *assms by fastforce*
 ultimately have *bounded ?f by (blast intro: boundedI)*
 hence *False using bounded-iff-finite-range assms by force*
 }
 thus *?thesis using that by fastforce*
 qed

lemma *bounded-odd*:
 fixes *f :: int \Rightarrow int*
 assumes $\bigwedge z. z < 0 \implies f z = -f (-z) \bigwedge n. n > 0 \implies |f n| \leq C$
 shows *bounded f*
proof –
 have $|f n| \leq C + |f 0|$ if $n \geq 0$ for n using *assms by (metis abs-ge-zero abs-of-nonneg add-increasing2 le-add-same-cancel2 that zero-less-abs-iff)*
 hence $|f n| \leq C + |f 0|$ for n using *assms by (cases $0 \leq n$) fastforce+*
 thus *?thesis by (rule boundedI)*
 qed

lemma *slope-odd*:
 assumes $\bigwedge z. z < 0 \implies f z = -f (-z)$
 $\bigwedge m n. \llbracket m > 0; n > 0 \rrbracket \implies |f (m + n) - (f m + f n)| \leq C$
 shows *slope f*
proof –
 define *C' where $C' = C + |f 0|$*
 have $C \geq 0$ using *assms(2)[of 1 1] by simp*
 hence *bound: $|f (m + n) - (f m + f n)| \leq C'$ if $m \geq 0$ $n \geq 0$ for $m n$*
 unfolding *C'-def using assms(2) that*


```

by (cases m = 0 ∨ n = 0) (force, metis abs-ge-zero add-increasing2 order-le-less)
{
  fix m n
  have |f (m + n) - (f m + f n)| ≤ C'
  proof (cases m ≥ 0)
    case m-nonneg: True
    show ?thesis
    proof (cases n ≥ 0)
      case True
      thus ?thesis using bound m-nonneg by fast
    next
      case False
      hence f-n: f n = - f (- n) using assms by simp
      show ?thesis
      proof (cases m + n ≥ 0)
        case True
        have |f (m + n) - (f m + f n)| = |f (m + n + -n) - (f (m + n) + f
(-n))| using f-n by auto
        thus ?thesis using bound[OF True] by (metis False neg-0-le-iff-le nle-le)
      next
        case False
        hence f (m + n) = - f (- (m + n)) using assms by force
        hence |f (m + n) - (f m + f n)| = |f (- (m + n) + m) - (f (- (m +
n)) + f m)| using f-n by force
        thus ?thesis using m-nonneg bound[of - (m + n) m] False by simp
      qed
    qed
  next
    case m-neg: False
    hence f-m: f m = - f (- m) using assms by simp
    show ?thesis
    proof (cases n ≥ 0)
      case True
      show ?thesis
      proof (cases m + n ≥ 0)
        case True
        have |f (m + n) - (f m + f n)| = |f (m + n + -m) - (f (m + n) + f
(-m))| using f-m by force
        thus ?thesis using bound[OF True, of - m] m-neg by simp
      next
        case False
        hence f (m + n) = - f (- (m + n)) using assms by force
        hence |f (m + n) - (f m + f n)| = |f (- (m + n) + n) - (f (- (m +
n)) + f n)| using f-m by force
        thus ?thesis using bound[of - (m + n) n] True False by simp
      qed
    next
      case False
      hence f-n: f n = - f (- n) using assms by simp

```

have $f(m + n) = -f(-m + -n)$ **using** *m-neg False assms* **by** *fastforce*
hence $|f(m + n) - (fm + fn)| = |-f(-m + -n) - (-f(-m) + -f(-n))|$ **using** *f-m f-n* **by** *argo*
also have $\dots = |f(-m + -n) - (f(-m) + f(-n))|$ **by** *linarith*
finally show *?thesis* **using** *bound[of - m - n] False m-neg* **by** *simp*
qed
qed
}
thus *?thesis* **unfolding** *slope-def* **by** (*fast intro: boundedI*)
qed

lemma *slope-bounded-comp-right-abs:*

assumes *slope f bounded (f o abs)*

shows *bounded f*

proof –

obtain B **where** $\forall z. |f z| \leq B$ **and** *B-nonneg: $0 \leq B$* **using** *assms* **by** *fastforce*

hence *B-bound: $\forall z \geq 0. |f z| \leq B$* **by** (*metis abs-of-nonneg*)

obtain D **where** *D-bound: $|f(m + n) - (fm + fn)| \leq D$* **and** *D-nonneg: $0 \leq D$* **for** $m n$ **using** *assms* **by** *fast*

have *bound: $|f(-m)| \leq |f 0| + B + D$* **if** $m \geq 0$ **for** m **using** *D-bound[of -m m] B-bound* **that** **by** *auto*

have $|f z| \leq |f 0| + B + D$ **for** z **using** *B-bound B-nonneg D-nonneg bound[of -z]* **by** (*cases $z \geq 0$*) *fastforce+*

thus *bounded f* **by** (*rule boundedI*)

qed

corollary *slope-finite-range-iff:*

assumes *slope f*

shows *finite (range f) \longleftrightarrow finite (f ‘ {0..})* (**is** *?lhs \longleftrightarrow ?rhs*)

proof (*rule iffI*)

assume *asm: ?rhs*

have *range (f o abs) = f ‘ {0..}* **unfolding** *comp-def atLeast-def image-def* **by** (*metis UNIV-I abs-ge-zero abs-of-nonneg mem-Collect-eq*)

thus *?lhs* **using** *slope-bounded-comp-right-abs[OF assms]* *asm* **by** (*fastforce simp add: bounded-iff-finite-range*)

qed (*metis image-subsetI rangeI finite-subset*)

lemma *slope-positive-lower-bound:*

assumes *slope f infinite (f ‘ {0..} \cap {0<..}) $D > 0$*

obtains M **where** $M > 0 \wedge m. m > 0 \implies (m + 1) * D \leq f(m * M)$

proof –

{

have *D-nonneg: $D \geq 0$* **using** *assms* **by** *force*

obtain C **where** *C-bound: $|f(m + n) - (fm + fn)| \leq C$* **and** *C-nonneg: $0 \leq C$* **for** $m n$ **using** *assms* **by** *fast*

obtain $f \cdot M$ **where** $2 * (C + D) \leq |f \cdot M|$ $f \cdot M \in (f \cdot \{0..\} \cap \{0<..\})$ **using** *mult-left-mono*[of $C + D - 2$] *D-nonneg* **by** (*metis* *assms*(2) *abs-ge-zero* *abs-le-D1* *int-set-infiniteD*)

then obtain M **where** M -bound: $2 * (C + D) \leq |f \cdot M|$ $0 < f \cdot M$ **and** M -nonneg: $0 \leq M$ **by** *blast*

have *neg-bound*: $(f (m * M + M) - (f (m * M) + f M)) \geq -C$ **for** m **by** (*metis* *C-bound* *abs-diff-le-iff* *minus-int-code*(1,2))

hence *neg-bound'*: $(f (m * M + M) - (f (m * M) + f M)) \geq -(C + D)$ **for** m **by** (*meson* *D-nonneg* *add-increasing2* *minus-le-iff*)

have $*$: $m > 0 \implies f (m * M) \geq (m + 1) * (C + D)$ **for** m

proof (*induction* m *rule*: *int-induct*[**where** $?k=1$])

case *base*

show $?case$ **using** M -bound **by** *fastforce*

next

case (*step1* m)

have $(m + 1 + 1) * (C + D) = (m + 1) * (C + D) + 2 * (C + D) - (C + D)$ **by** *algebra*

also have $\dots \leq (m + 1) * (C + D) + f \cdot M + -(C + D)$ **using** M -bound **by** *fastforce*

also have $\dots \leq f (m * M) + f \cdot M + -(C + D)$ **using** *step1* **by** *simp*

also have $\dots \leq (f (m * M) + f \cdot M) + (f (m * M + M) - (f (m * M) + f \cdot M))$ **using** *add-left-mono*[*OF* *neg-bound'*] **by** *blast*

also have $\dots = f ((m + 1) * M)$ **by** (*simp* *add*: *distrib-right*)

finally show $?case$ **by** *blast*

next

case (*step2* i)

then show $?case$ **by** *linarith*

qed

have $*$: $f (m * M) \geq (m + 1) * D$ **if** $m > 0$ **for** m **using** $*$ [*OF* *that*] *mult-left-mono*[of D $C + D$ $m + 1$] *that* *C-nonneg* *D-nonneg* **by** *linarith*

moreover have $M \neq 0$ **using** M -bound *add1-zle-eq* *assms* *neg-bound* **by** *force*

ultimately have $\exists M > 0. \forall m > 0. (m + 1) * D \leq f (m * M)$ **using** M -nonneg **by** *force*

}

thus $?thesis$ **using** *that* **by** *blast*

qed

1.3 Set Membership of *Inf* and *Sup* on Integers

lemma *int-Inf-mem*:

fixes $S :: \text{int set}$

assumes $S \neq \{\}$ *bdd-below* S

shows $\text{Inf } S \in S$

proof –

have *nonneg*: $\text{Inf } (\{0..\} \cap A) \in (\{0..\} \cap A)$ **if** *asm*: $(\{(0::\text{int})..\} \cap A) \neq \{\}$ **for** A

proof –
have $\text{nat } ' (\{0..\} \cap A) \neq \{\}$ **using** *asm* **by** *blast*
hence $\text{int } (\text{Inf } (\text{nat } ' (\{0..\} \cap A))) \in \text{int } ' \text{nat } ' (\{0..\} \cap A)$ **using** *wellorder-InfI*[of
- $\text{nat } ' (\{0..\} \cap A)$] **by** *fast*
moreover **have** $\text{int } ' \text{nat } ' (\{0..\} \cap A) = \{0..\} \cap A$ **by** *force*
moreover **have** $\text{Inf } (\{0..\} \cap A) = \text{int } (\text{Inf } (\text{nat } ' (\{0..\} \cap A)))$
using *calculation* **by** (*intro cInf-eq-minimum*) (*argo*, *metis IntD2 Int-commute*
atLeast-iff imageI le-nat-iff wellorder-Inf-le1)
ultimately show *?thesis* **by** *argo*
qed
have ******: $\text{Inf } (\{b..\} \cap A) \in (\{b..\} \cap A)$ **if** *asm*: $(\{(b::\text{int})..\} \cap A) \neq \{\}$ **for** A b
proof (*cases* $b \geq 0$)
case *True*
hence $(\{b..\} \cap A) = \{0..\} \cap (\{b..\} \cap A)$ **by** *fastforce*
thus *?thesis* **using** *asm nonneg* **by** *metis*
next
case *False*
hence *partition*: $(\{b..\} \cap A) = (\{0..\} \cap A) \cup (\{b..<0\} \cap A)$ **by** *fastforce*
have *bdd-below*: $\text{bdd-below } (\{0..\} \cap A) \text{ bdd-below } (\{b..<0\} \cap A)$ **by** *simp+*
thus *?thesis*
proof (*cases* $(\{0..\} \cap A) \neq \{\} \wedge (\{b..<0\} \cap A) \neq \{\}$)
case *True*
have *finite*: $\text{finite } (\{b..<0\} \cap A)$ **by** *blast*
have $(x :: \text{int}) \leq y \implies \text{inf } x \ y = x$ **for** $x \ y$ **by** (*simp add: inf.order-iff*)
have $\text{Inf } (\{b..\} \cap A) = \text{inf } (\text{Inf } (\{0..\} \cap A)) (\text{Inf } (\{b..<0\} \cap A))$ **by** (*metis*
cInf-union-distrib True bdd-below partition)
moreover **have** $\text{Inf } (\{b..<0\} \cap A) \in (\{b..\} \cap A)$ **using** *Min-in*[*OF finite*]
cInf-eq-Min[*OF finite*] *True partition* **by** *simp*
moreover **have** $\text{Inf } (\{0..\} \cap A) \in (\{b..\} \cap A)$ **using** *nonneg True partition*
by *blast*
moreover **have** $\text{inf } (\text{Inf } (\{0..\} \cap A)) (\text{Inf } (\{b..<0\} \cap A)) \in \{\text{Inf } (\{0..\}$
 $\cap A), \text{Inf } (\{b..<0\} \cap A)\}$ **by** (*metis inf.commute inf.order-iff insertI1 insertI2*
nle-le)
ultimately show *?thesis* **by** *force*
next
case *False*
hence $(\{b..\} \cap A) = (\{0..\} \cap A) \vee (\{b..\} \cap A) = (\{b..<0\} \cap A)$ **using**
partition **by** *auto*
thus *?thesis* **using** *Min-in*[of $\{b..\} \cap A$] *cInf-eq-Min*[of $\{b..\} \cap A$] **by** (*metis*
asm nonneg finite-Int finite-atLeastLessThan-int)
qed
qed
obtain b **where** $S = \{b..\} \cap S$ **using** *assms unfolding bdd-below-def* **by** *blast*
thus *?thesis* **using** ****** *assms* **by** *metis*
qed

lemma *int-Sup-mem*:
fixes $S :: \text{int set}$
assumes $S \neq \{\}$ *bdd-above* S

```

  shows  $Sup\ S \in S$ 
proof –
  have  $Sup\ S = (-\ Inf\ (uminus\ 'S))$  unfolding Inf-int-def image-comp by simp
  moreover have  $bdd\ below\ (uminus\ 'S)$  using assms unfolding bdd-below-def
bdd-above-def by (metis imageE neg-le-iff-le)
  moreover have  $Inf\ (uminus\ 'S) \in (uminus\ 'S)$  using int-Inf-mem assms by
simp
  ultimately show ?thesis by force
qed

end

```

```

theory Eudoxus
  imports Slope
begin

```

2 Eudoxus Reals

2.1 Type Definition

Two slopes are said to be equivalent if their difference is bounded.

definition *eudoxus-rel* :: $(int \Rightarrow int) \Rightarrow (int \Rightarrow int) \Rightarrow bool$ (**infix** \sim_e 50) **where**

$$f \sim_e g \equiv slope\ f \wedge slope\ g \wedge bounded\ (\lambda n. f\ n - g\ n)$$

```

lemma eudoxus-rel-equivp:
  part-equivp eudoxus-rel
proof (rule part-equivpI)
  show  $\exists x. x \sim_e x$  unfolding eudoxus-rel-def slope-def bounded-def by fast
  show symp  $(\sim_e)$  unfolding eudoxus-rel-def by (force intro: sympI dest: bounded-uminus
simp: fun-Compl-def)
  show transp  $(\sim_e)$  unfolding eudoxus-rel-def by (force intro!: transpI dest:
bounded-add)
qed

```

We define the reals as the set of all equivalence classes of the relation (\sim_e) .

```

quotient-type real =  $(int \Rightarrow int) / partial: eudoxus-rel$ 
  by (rule eudoxus-rel-equivp)

```

```

lemma real-quot-type: quot-type  $(\sim_e)$  Abs-real Rep-real
  using Rep-real Abs-real-inverse Rep-real-inverse Rep-real-inject eudoxus-rel-equivp
by (auto intro!: quot-type.intro)

```

```

lemma slope-refl:  $slope\ f = (f \sim_e f)$ 
  unfolding eudoxus-rel-def by (fastforce simp add: bounded-constant)

```

```

declare slope-refl[THEN iffD2, simp]

```

lemmas *slope-refI* = *slope-refl*[*THEN iffD1*]

lemma *slope-induct*[*consumes 0, case-names slope*]:

assumes $\bigwedge f. \text{slope } f \implies P (\text{abs-real } f)$

shows $P x$

using *assms* **by** *induct force*

lemma *abs-real-eq-iff*: $f \sim_e g \iff \text{slope } f \wedge \text{slope } g \wedge \text{abs-real } f = \text{abs-real } g$

by (*metis Quotient-real Quotient-rel slope-refl*)

lemma *abs-real-eqI*[*intro*]: $f \sim_e g \implies \text{abs-real } f = \text{abs-real } g$ **using** *abs-real-eq-iff*

by *blast*

lemmas *eudoxus-rel-sym*[*sym*] = *Quotient-symp*[*OF Quotient-real, THEN sympD*]

lemmas *eudoxus-rel-trans*[*trans*] = *Quotient-transp*[*OF Quotient-real, THEN transpD*]

lemmas *rep-real-abs-real-refl* = *Quotient-rep-abs*[*OF Quotient-real, OF slope-refl*][*THEN iffD1, intro!*]

lemmas *rep-real-iff* = *Quotient-rel-rep*[*OF Quotient-real, iff*]

declare *Quotient-abs-rep*[*OF Quotient-real, simp*]

lemma *slope-rep-real*: *slope (rep-real x)* **by** *simp*

lemma *eudoxus-relI*:

assumes $\text{slope } f \text{ slope } g \wedge n. n \geq N \implies |f n - g n| \leq C$

shows $f \sim_e g$

proof –

have *C-nonneg*: $C \geq 0$ **using** *assms* **by** *force*

obtain *C-f* **where** *C-f*: $|f (n + (- n)) - (f n + f (- n))| \leq C-f \ 0 \leq C-f$ **for** n **using** *assms* **by** *fast*

obtain *C-g* **where** *C-g*: $|g (n + (- n)) - (g n + g (- n))| \leq C-g \ 0 \leq C-g$ **for** n **using** *assms* **by** *fast*

have *bound*: $|f (- n) - g (- n)| \leq |f n - g n| + |f 0| + |g 0| + C-f + C-g$ **for** n **using** *C-f(1)[of n]* *C-g(1)[of n]* **by** *simp*

define *C'* **where** $C' = \text{Sup } \{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\} + C + |f 0| + |g 0| + C-f + C-g$

have *: *bdd-above* $\{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\}$ **by** *simp*

have *Sup* $\{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\} \in \{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\}$ **using** *C-nonneg* **by** (*intro int-Sup-mem*[*OF - **]) *auto*

hence *Sup-nonneg*: $\text{Sup } \{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\} \geq 0$ **by** *fastforce*

have *: $|f n - g n| \leq \text{Sup } \{|f n - g n| \mid n. n \in \{0..max \ 0 \ N\}\} + C$ **if** $n \geq 0$ **for** n **unfolding** *C'-def* **using** *cSup-upper*[*OF - **] *that C-nonneg Sup-nonneg* **by**

```

(cases  $n \leq N$ ) (fastforce simp add: add commute add-increasing2 assms(3))+
{
  fix  $n$ 
  have  $|f\ n - g\ n| \leq C'$ 
  proof (cases  $n \geq 0$ )
    case True
      thus ?thesis unfolding C'-def using * C-f C-g by fastforce
    next
      case False
        hence  $-n \geq 0$  by simp
        hence  $|f\ (-n) - g\ (-n)| \leq \text{Sup } \{|f\ n - g\ n| \mid n. n \in \{0..max\ 0\ N\}\} + C$ 
  using *[of  $-n$ ] by blast
  hence  $|f\ (-(-n)) - g\ (-(-n))| \leq C'$  unfolding C'-def using bound[of  $-n$ ] by linarith
  thus ?thesis by simp
qed
}
thus ?thesis using assms unfolding eudoxus-rel-def by (auto intro: boundedI)
qed

```

2.2 Addition and Subtraction

We define addition, subtraction and the additive identity as follows.

```

instantiation real :: {zero, plus, minus, uminus}
begin

```

quotient-definition

```

  0 :: real is abs-real ( $\lambda-. 0$ ) .

```

```

declare slope-zero[intro!, simp]

```

```

lemma zero-iff-bounded:  $f \sim_e (\lambda-. 0) \iff$  bounded  $f$  by (metis (no-types, lifting)
boundedE boundedI diff-zero eudoxus-rel-def slope-zero bounded-slopeI)

```

```

lemma zero-iff-bounded':  $x = 0 \iff$  bounded (rep-real  $x$ ) by (metis (mono-tags)
abs-real-eq-iff id-apply rep-real-abs-real-refl rep-real-iff slope-zero zero-iff-bounded
zero-real-def)

```

```

lemma zero-def:  $0 =$  abs-real ( $\lambda-. 0$ ) unfolding zero-real-def by simp

```

```

definition eudoxus-plus :: ( $int \Rightarrow int$ )  $\Rightarrow$  ( $int \Rightarrow int$ )  $\Rightarrow$  ( $int \Rightarrow int$ ) (infixl  $+_e$ 
60) where

```

```

  ( $f :: int \Rightarrow int$ )  $+_e$   $g = (\lambda z. f\ z + g\ z)$ 

```

```

declare slope-add[intro, simp]

```

quotient-definition

```

  (+) :: ( $real \Rightarrow real \Rightarrow real$ ) is ( $+_e$ )

```

proof –

```

  fix  $x\ x'\ y\ y'$  assume  $x \sim_e x'\ y \sim_e y'$ 

```

hence *rel-x: slope x slope x' bounded* $(\lambda z. x z - x' z)$ **and** *rel-y: slope y slope y' bounded* $(\lambda z. y z - y' z)$ **unfolding** *eudoxus-rel-def* **by** *blast+*
thus $(x +_e y) \sim_e (x' +_e y')$ **unfolding** *eudoxus-rel-def eudoxus-plus-def* **by**
(fastforce intro: back-subst[of bounded, OF bounded-add[OF rel-x(\mathcal{I}) rel-y(\mathcal{I})]])
qed

lemmas *eudoxus-plus-cong* = *apply-rsp'[OF plus-real.rsp, THEN rel-funD, intro]*

lemma *abs-real-plus[simp]*:
assumes *slope f slope g*
shows *abs-real f + abs-real g = abs-real (f +_e g)*
using *assms unfolding plus-real-def* **by** *auto*

definition *eudoxus-uminus* :: $(int \Rightarrow int) \Rightarrow (int \Rightarrow int) (-_e)$ **where**
 $-_e (f :: int \Rightarrow int) = (\lambda x. - f x)$

declare *slope-uminus'[intro, simp]*

quotient-definition

(uminus) :: (real \Rightarrow real) is $-_e$
proof $-$
fix $x x'$ **assume** $x \sim_e x'$
hence *rel-x: slope x slope x' bounded* $(\lambda z. x z - x' z)$ **unfolding** *eudoxus-rel-def*
by *blast+*
thus $-_e x \sim_e -_e x'$ **unfolding** *eudoxus-rel-def eudoxus-uminus-def* **by** *(fastforce intro: back-subst[of bounded, OF bounded-uminus[OF rel-x(\mathcal{I})]])*
qed

lemmas *eudoxus-uminus-cong* = *apply-rsp'[OF uminus-real.rsp, simplified, intro]*

lemma *abs-real-uminus[simp]*:
assumes *slope f*
shows $- abs-real f = abs-real (-_e f)$
using *assms unfolding uminus-real-def* **by** *auto*

definition $x - (y :: real) = x + - y$

declare *slope-minus[intro, simp]*

lemma *abs-real-minus[simp]*:
assumes *slope g slope f*
shows $abs-real g - abs-real f = abs-real (g +_e (-_e f))$
using *assms by (simp add: minus-real-def slope-refl eudoxus-uminus-cong)*

instance ..
end

The Eudoxus reals equipped with addition and negation specified as above constitute an Abelian group.


```

instance real :: ab-group-add
proof
  fix x y z :: real
  show x + y + z = x + (y + z) by (induct x, induct y, induct z) (simp add:
eudoxus-plus-cong eudoxus-plus-def add.assoc)
  show x + y = y + x by (induct x, induct y) (simp add: eudoxus-plus-def
add.commute)
  show 0 + x = x by (induct x) (simp add: zero-real-def eudoxus-plus-def)
  show - x + x = 0 by (induct x) (simp add: eudoxus-uminus-cong, simp add:
zero-real-def eudoxus-plus-def eudoxus-uminus-def)
qed (simp add: minus-real-def)

```

2.3 Multiplication

We define multiplication as the composition of two slopes.

```

instantiation real :: {one, times}
begin

quotient-definition
  1 :: real is abs-real id .

declare slope-one[intro!, simp]

lemma one-def: 1 = abs-real id unfolding one-real-def by simp

definition eudoxus-times :: (int  $\Rightarrow$  int)  $\Rightarrow$  (int  $\Rightarrow$  int)  $\Rightarrow$  int  $\Rightarrow$  int (infixl *_e 60)
where
  f *_e g = f o g

declare slope-comp[intro, simp]
declare slope-scale[intro, simp]

quotient-definition
  (*) :: real  $\Rightarrow$  real  $\Rightarrow$  real is (*_e)
proof -
  fix x x' y y' assume x  $\sim_e$  x' y  $\sim_e$  y'
  hence rel-x: slope x slope x' bounded ( $\lambda z. x z - x' z$ ) and rel-y: slope y slope y'
bounded ( $\lambda z. y z - y' z$ ) unfolding eudoxus-rel-def by blast+

  obtain C where x'-bound:  $|x' (m + n) - (x' m + x' n)| \leq C$  for m n using
rel-x(2) unfolding slope-def by fastforce

  obtain A B where x'-lin-bound:  $|x' n| \leq A * |n| + B$   $0 \leq A$   $0 \leq B$  for n using
slope-linear-bound[OF rel-x(2)] by blast

  obtain C' where y-y'-bound:  $|y z - y' z| \leq C'$  for z using rel-y(3) unfolding
slope-def by fastforce

  have bounded ( $\lambda z. x' (y z) - x' (y' z)$ )

```

proof (*rule boundedI*)
fix z
have $|x'(y z) - x'(y' z)| \leq |x'(y z - y' z)| + C$ **using** x' -*bound*[*of* $y z - y'$
 $z y' z$] **by** *fastforce*
also have $\dots \leq A * |y z - y' z| + B + C$ **using** x' -*lin-bound* **by** *force*
also have $\dots \leq A * C' + B + C$ **using** *mult-left-mono*[*OF* y - y' -*bound* x' -*lin-bound*(2)]
by *fastforce*
finally show $|x'(y z) - x'(y' z)| \leq A * C' + B + C$ **by** *blast*
qed
hence *bounded* ($\lambda z. x (y z) - x' (y' z)$) **using** *bounded-add*[*OF* *bounded-comp*(1)[*OF*
rel-x(β), *of* y]] **by** *force*
thus $(x *_e y) \sim_e (x' *_e y')$ **unfolding** *eudoxus-rel-def* *eudoxus-times-def* **using**
rel-x *rel-y* **by** *simp*
qed

lemmas *eudoxus-times-cong* = *apply-rsp*'[*OF* *times-real.rsp*, *THEN* *rel-funD*, *intro*]
lemmas *eudoxus-rel-comp* = *eudoxus-times-cong*[*unfolded* *eudoxus-times-def*]

lemma *eudoxus-times-commute*:
assumes *slope* f *slope* g
shows $(f *_e g) \sim_e (g *_e f)$
unfolding *eudoxus-rel-def* *eudoxus-times-def*
using *slope-comp* *slope-comp-commute* *assms* **by** *blast*

lemma *abs-real-times[simp]*:
assumes *slope* f *slope* g
shows *abs-real* $f * \text{abs-real } g = \text{abs-real } (f *_e g)$
using *assms* **unfolding** *times-real-def* **by** *auto*

instance ..
end

lemma *neg-one-def*: $- 1 = \text{abs-real } (-_e \text{id})$ **unfolding** *one-real-def* **by** (*simp* *add*:
eudoxus-uminus-def)
lemma *slope-neg-one*[*intro*, *simp*]: *slope* $(-_e \text{id})$ **using** *slope-refl* **by** *blast*

With the definitions provided above, the Eudoxus reals are a commutative ring with unity.

instance *real* :: *comm-ring-1*
proof

fix $x y z$:: *real*
show $x * y * z = x * (y * z)$ **by** (*induct* x , *induct* y , *induct* z) (*simp* *add*:
eudoxus-times-cong *eudoxus-times-def* *comp-assoc*)
show $x * y = y * x$ **by** (*induct* x , *induct* y) (*force* *simp* *add*: *slope-refl* *eudoxus-times-commute*)
show $1 * x = x$ **by** (*induct* x) (*simp* *add*: *one-real-def* *eudoxus-times-def*)
show $(x + y) * z = x * z + y * z$ **by** (*induct* x , *induct* y , *induct* z) (*simp* *add*: *eudoxus-times-cong* *eudoxus-plus-cong*, *simp* *add*: *eudoxus-times-def* *eudoxus-plus-def*)

comp-def)
have \neg bounded $(\lambda x. x)$ **by** (*metis add.inverse-inverse boundedE-strict less-irrefl neg-less-0-iff-less zabs-def*)
thus $(0 :: \text{real}) \neq (1 :: \text{real})$ **using** *abs-real-eq-iff*[*of id $\lambda \cdot. 0$*] **unfolding** *one-real-def zero-real-def eudoxus-rel-def* **by** *simp*
qed

lemma *real-of-nat*:
 $\text{of-nat } n = \text{abs-real } ((*) (\text{of-nat } n))$
proof (*induction n*)
case 0
then show *?case* **by** (*simp add: zero-real-def*)
next
case (*Suc n*)
then show *?case* **by** (*simp add: one-real-def distrib-right eudoxus-plus-def*)
qed

lemma *real-of-int*:
 $\text{of-int } z = \text{abs-real } ((*) z)$
proof (*induction z rule: int-induct*[**where** *?k=0*])
case *base*
then show *?case* **by** (*simp add: zero-real-def*)
next
case (*step1 i*)
then show *?case* **by** (*simp add: one-real-def distrib-right eudoxus-plus-def*)
next
case (*step2 i*)
then show *?case* **by** (*simp add: one-real-def eudoxus-plus-def left-diff-distrib eudoxus-uminus-def*)
qed

The Eudoxus reals are a ring of characteristic $0 :: 'a$.

instance *real* :: *ring-char-0*
proof
show *inj* $(\lambda n. \text{of-nat } n :: \text{real})$
proof (*intro inj-onI*)
fix $x y$ **assume** $(\text{of-nat } x :: \text{real}) = \text{of-nat } y$
hence $((*) (\text{int } x)) \sim_e ((*) (\text{int } y))$ **unfolding** *abs-real-eq-iff real-of-nat* **using** *slope-scale* **by** *blast*
hence bounded $(\lambda z. (\text{int } x - \text{int } y) * z)$ **unfolding** *eudoxus-rel-def* **by** (*simp add: left-diff-distrib*)
then obtain C **where** *bound*: $|(\text{int } x - \text{int } y) * z| \leq C$ **and** *C-nonneg*: $0 \leq C$ **for** z **by** *blast*
hence $|\text{int } x - \text{int } y| * |C + 1| \leq C$ **using** *abs-mult* **by** *metis*
hence $*$: $|\text{int } x - \text{int } y| * (C + 1) \leq C$ **using** *C-nonneg* **by** *force*
thus $x = y$ **using** *order-trans*[*OF mult-right-mono *, of 1*] *C-nonneg* **by** *fastforce*
qed
qed

2.4 Ordering

We call a slope positive, if it tends to infinity. Similarly, we call a slope negative if it tends to negative infinity.

instantiation $real :: \{ord, abs, sgn\}$
begin

definition $pos :: (int \Rightarrow int) \Rightarrow bool$ **where**
 $pos\ f = (\forall C \geq 0. \exists N. \forall n \geq N. f\ n \geq C)$

definition $neg :: (int \Rightarrow int) \Rightarrow bool$ **where**
 $neg\ f = (\forall C \geq 0. \exists N. \forall n \geq N. f\ n \leq -C)$

lemma $pos\ neg\ exclusive: \neg (pos\ f \wedge neg\ f)$ **unfolding** $neg\ def\ pos\ def$ **by** $(metis\ int\ one\ le\ iff\ zero\ less\ linorder\ not\ less\ nle\ le\ uminus\ int\ code(1)\ zero\ less\ one\ class.\ zero\ le\ one)$

lemma $pos\ iff\ neg\ uminus: pos\ f = neg\ (-_e\ f)$ **unfolding** $neg\ def\ pos\ def\ eu\ doxus\ uminus\ def$ **by** $simp$

lemma $neg\ iff\ pos\ uminus: neg\ f = pos\ (-_e\ f)$ **unfolding** $neg\ def\ pos\ def\ eu\ doxus\ uminus\ def$ **by** $fastforce$

lemma $pos\ iff:$

assumes $slope\ f$

shows $pos\ f = infinite\ (f\ \{0..\} \cap \{0<..})$ **(is** $?lhs = ?rhs$ **)**

proof $(rule\ iffI)$

assume $pos: ?lhs$

{

fix C **assume** $C\ nonneg: 0 \leq (C :: int)$

hence $\exists z \geq 0. (C + 1) \leq f\ z$ **by** $(metis\ add\ increasing2\ nle\ le\ zero\ less\ one\ class.\ zero\ le\ one\ pos\ pos\ def)$

hence $\exists z \geq 0. C \leq f\ z \wedge 0 < f\ z$ **using** $C\ nonneg$ **by** $fastforce$

hence $\exists N \geq C. \exists z. N = f\ z \wedge 0 < f\ z \wedge 0 \leq z$ **by** $blast$

}

thus $?rhs$ **by** $(blast\ intro!: int\ set\ infiniteI)$

next

assume $infinite: ?rhs$

then obtain D **where** $D\ bound: |f\ (m + n) - (f\ m + f\ n)| < D\ 0 < D$ **for** $m\ n$ **using** $assms$ **by** $(fastforce\ simp: slope\ def\ elim: boundedE\ strict)$

obtain M **where** $M\ bound: \forall m > 0. (m + 1) * D \leq f\ (m * M)\ 0 < M$ **using** $slope\ positive\ lower\ bound[OF\ assms\ infinite]$ $D\ bound(2)$ **by** $blast$

define g **where** $g = (\lambda z. f\ ((z\ div\ M) * M))$

define E **where** $E = Sup\ ((abs\ o\ f)\ \{z. 0 \leq z \wedge z < M\})$

have $E\ bound: |f\ (z\ mod\ M)| \leq E$ **for** z

proof $-$

have $(z\ mod\ M) \in \{z. 0 \leq z \wedge z < M\}$ **by** $(simp\ add: M\ bound(2))$

hence $|f(z \bmod M)| \in (\text{abs } o f) \text{ ' } \{z. 0 \leq z \wedge z < M\}$ **by** *fastforce*
thus $|f(z \bmod M)| \leq E$ **unfolding** *E-def* **by** (*simp add: le-cSup-finite*)
qed
hence *E-nonneg*: $0 \leq E$ **by** *fastforce*

have *diff-bound*: $|fz - gz| \leq E + D$ **for** z
proof –
let $?d = z \text{ div } M$ **and** $?r = z \text{ mod } M$
have *z-is*: $z = ?d * M + ?r$ **by** *presburger*
hence $|fz - gz| = |f(?d * M + ?r) - g(?d * M + ?r)|$ **by** *argo*
also have $\dots = |(f(?d * M + ?r) - (f(?d * M) + f ?r)) + (f(?d * M) + f ?r) - g(?d * M + ?r)|$ **by** *auto*
also have $\dots = |f ?r + (f(?d * M + ?r) - (f(?d * M) + f ?r))|$ **unfolding** *g-def* **by** *force*
also have $\dots \leq |f ?r| + D$ **using** *D-bound(1)*[*of ?d * M ?r*] **by** *linarith*
also have $\dots \leq E + D$ **using** *E-bound* **by** *simp*
finally show $|fz - gz| \leq E + D$.
qed
{
fix C **assume** *C-nonneg*: $0 \leq (C :: \text{int})$

define n **where** $n = (E + D + C) \text{ div } D$
hence *zero-less-n*: $n > 0$ **using** *D-bound(2)* *E-nonneg* *C-nonneg* **using** *pos-imp-zdiv-pos-iff*
by *fastforce*

have $E + C < E + D + C - (E + D + C) \bmod D$ **using** *diff-strict-left-mono*[*OF pos-mod-bound*[*OF D-bound(2)*]] **by** *simp*
also have $\dots = n * D$ **unfolding** *n-def* **using** *div-mod-decomp-int*[*of E + D + C D*] **by** *algebra*
finally have $*(n + 1) * D > E + D + C$ **by** (*simp add: add.commute distrib-right*)

have $C \leq f m$ **if** $m \geq n * M$ **for** m
proof –
let $?d = m \text{ div } M$ **and** $?r = m \text{ mod } M$
have *d-pos*: $?d > 0$ **using** *zero-less-n* *M-bound* *that dual-order.trans pos-imp-zdiv-pos-iff*
by *fastforce*
have *n-le-d*: $?d \geq n$ **using** *zdiv-mono1* *M-bound* *that* **by** *fastforce*
have $E + D + C < (?d + 1) * D$ **using** *D-bound* *n-le-d* **by** (*intro* *[*THEN order.strict-trans2*]) *simp*
also have $\dots \leq g m$ **unfolding** *g-def* **using** *M-bound* *d-pos* **by** *blast*
finally have $E + D + C < g m$.
hence $|f m - g m| + C < g m$ **using** *diff-bound*[*of m*] **by** *fastforce*
thus *?thesis* **by** *fastforce*
qed
hence $\exists N. \forall p \geq N. C \leq f p$ **using** *add1-zle-eq* **by** *blast*

}
thus *?lhs* **unfolding** *pos-def* **by** *blast*
qed

lemma *neg-iff*:
assumes *slope f*
shows $\text{neg } f = \text{infinite } (f \text{ ‘ } \{0..\} \cap \{..\lt 0\})$ (**is** *?lhs = ?rhs*)
proof (*rule iffI*)
assume *?lhs*
hence $\text{infinite } ((- f) \text{ ‘ } \{0..\} \cap \{0<..\})$ **using** *pos-iff[OF slope-uminus'[OF assms]]* **unfolding** *neg-def pos-def* **by** *fastforce*
moreover **have** *inj (uminus :: int \Rightarrow int)* **by** *simp*
moreover **have** $(- f) \text{ ‘ } \{0..\} \cap \{0<..\} = \text{uminus ‘ } (f \text{ ‘ } \{0..\} \cap \{..\lt 0\})$ **by** *fastforce*
ultimately **show** *?rhs* **using** *finite-imageD* **by** *fastforce*
next
assume *?rhs*
moreover **have** *inj (uminus :: int \Rightarrow int)* **by** *simp*
moreover **have** $f \text{ ‘ } \{0..\} \cap \{..\lt 0\} = \text{uminus ‘ } ((- f) \text{ ‘ } \{0..\} \cap \{0<..\})$ **by** *force*
ultimately **have** $\text{infinite } ((- f) \text{ ‘ } \{0..\} \cap \{0<..\})$ **using** *finite-imageD* **by** *force*
thus *?lhs* **using** *pos-iff[OF slope-uminus'[OF assms]]* **unfolding** *pos-def neg-def* **by** *fastforce*
qed

lemma *pos-cong*:
assumes $f \sim_e g$
shows $\text{pos } f = \text{pos } g$
proof –
{
fix $x y$ **assume** *asm: pos x x \sim_e y*
fix D **assume** $D: 0 \leq D \forall N. \exists p \geq N. \neg D \leq y p$
obtain C **where** *bounds: $\forall n. |x n - y n| \leq C$ $0 \leq C$* **using** *asm* **unfolding** *eudoxus-rel-def* **by** *blast*
obtain N **where** $\forall p \geq N. C + D \leq x p$ **using** *D bounds asm* **by** (*fastforce simp add: pos-def*)
hence $\forall p \geq N. |x p - y p| + D \leq x p$ **by** (*metis add.commute add-left-mono bounds(1) dual-order.trans*)
hence $\forall p \geq N. D \leq y p$ **by** *force*
hence *False* **using** D **by** *blast*
}
hence $\text{pos } x \Longrightarrow \text{pos } y$ **if** $x \sim_e y$ **for** $x y$ **using** *that* **unfolding** *pos-def* **by** *metis*
thus *?thesis* **by** (*metis assms eudoxus-rel-equiv part-equivp-symp*)
qed

lemma *neg-cong*:
assumes $f \sim_e g$
shows $\text{neg } f = \text{neg } g$
proof –
{
fix $x y$ **assume** *asm: neg x x \sim_e y*
fix D **assume** $D: 0 \leq D \forall N. \exists p \geq N. \neg - D \geq y p$
obtain C **where** *bounds: $|x n - y n| \leq C$ $0 \leq C$* **for** n **using** *asm* **unfolding**

eudoxus-rel-def **by** *blast*
obtain N **where** $\forall p \geq N. - (C + D) \geq x p$ **using** *D bounds asm add-increasing2*
unfolding *neg-def* **by** *meson*
hence $\forall p \geq N. - |x p - y p| - D \geq x p$ **using** *bounds(1)[THEN le-imp-neg-le, THEN diff-right-mono, THEN dual-order.trans]* **by** *simp*
hence $\forall p \geq N. - D \geq y p$ **by** *force*
hence *False* **using** *D* **by** *blast*
}
hence $neg\ x \implies neg\ y$ **if** $x \sim_e y$ **for** $x\ y$ **using** *that unfolding neg-def* **by** *metis*
thus *?thesis* **by** (*metis assms eudoxus-rel-equivp part-equivp-symp*)
qed

lemma *pos-iff-nonneg-nonzero*:

assumes *slope f*
shows $pos\ f \longleftrightarrow (\neg\ neg\ f) \wedge (\neg\ bounded\ f)$ (**is** *?lhs* \longleftrightarrow *?rhs*)
proof (*rule iffI*)
assume *pos: ?lhs*
then obtain N **where** $\forall n \geq N. f\ n > 0$ **unfolding** *pos-def* **by** (*metis int-one-le-iff-zero-less zero-less-one-class.zero-le-one*)
hence $f\ (max\ N\ m) > 0$ **for** m **by** *simp*
hence $\neg\ neg\ f$ **unfolding** *neg-def* **by** (*metis add.inverse-neutral dual-order.refl linorder-not-le max.cobounded2*)
thus *?rhs* **using** *pos* **unfolding** *pos-def bounded-def bdd-above-def* **by** (*metis abs-ge-self dual-order.trans gt-ex imageI iso-tuple-UNIV-I order.strict-iff-not*)
next
assume *nonneg-nonzero: ?rhs*
hence *finite: finite* ($f\ ' \{0..\} \cap \{..<0\}$) **using** *neg-iff assms* **by** *blast*
moreover have *unbounded: infinite* ($f\ ' \{0..\}$) **using** *nonneg-nonzero bounded-iff-finite-range slope-finite-range-iff assms* **by** *blast*
ultimately have *infinite* ($f\ ' \{0..\} \cap \{0..\}$) **by** (*metis Compl-atLeast Diff-Diff-Int Diff-eq Diff-infinite-finite*)
moreover have $f\ ' \{0..\} \cap \{0<..\} = f\ ' \{0..\} \cap \{0..\} - \{0\}$ **by** *force*
ultimately show *?lhs* **unfolding** *pos-iff[OF assms]* **by** *simp*
qed

lemma *neg-iff-nonpos-nonzero*:

assumes *slope f*
shows $neg\ f \longleftrightarrow (\neg\ pos\ f) \wedge (\neg\ bounded\ f)$
unfolding *pos-iff-nonneg-nonzero[OF assms]* *neg-iff-pos-uminus uminus-apply eudoxus-uminus-def pos-iff-nonneg-nonzero[OF slope-uminus', OF assms]*
by (*force simp add: bounded-def bdd-above-def*)

We define the sign of a slope to be *id* if it is positive, $-_e\ id$ if it is negative and $\lambda\cdot\ 0::'b$ otherwise.

definition *eudoxus-sgn* $:: (int \Rightarrow int) \Rightarrow (int \Rightarrow int)$ **where**
eudoxus-sgn $f = (if\ pos\ f\ then\ id\ else\ if\ neg\ f\ then\ -_e\ id\ else\ (\lambda\cdot\ 0))$

lemma *eudoxus-sgn-iff*:

assumes *slope f*

shows $eudoxus\text{-}sgn\ f = (\lambda\cdot. 0) \longleftrightarrow bounded\ f$
 $eudoxus\text{-}sgn\ f = id \longleftrightarrow pos\ f$
 $eudoxus\text{-}sgn\ f = (-_e\ id) \longleftrightarrow neg\ f$
using $eudoxus\text{-}sgn\text{-}def\ neg\text{-}one\text{-}def\ one\text{-}def\ zero\text{-}def\ assms\ neg\text{-}iff\text{-}nonpos\text{-}nonzero$
 $pos\text{-}iff\text{-}nonneg\text{-}nonzero$ **by** $auto$

quotient-definition

$(sgn :: real \Rightarrow real)$ **is** $eudoxus\text{-}sgn$
unfolding $eudoxus\text{-}sgn\text{-}def$
using $eudoxus\text{-}uminus\text{-}cong\ neg\text{-}cong\ pos\text{-}cong\ slope\text{-}one\ slope\text{-}refl$ **by** $fastforce$

lemmas $eudoxus\text{-}sgn\text{-}cong = apply\text{-}rsp'[OF\ sgn\text{-}real.rsp,\ intro]$

lemma $eudoxus\text{-}sgn\text{-}cong'[cong]$:
assumes $f \sim_e g$
shows $eudoxus\text{-}sgn\ f = eudoxus\text{-}sgn\ g$
using $assms\ eudoxus\text{-}sgn\text{-}def\ neg\text{-}cong\ pos\text{-}cong$ **by** $presburger$

lemma $sgn\text{-}range$: $sgn\ (x :: real) \in \{-1, 0, 1\}$ **unfolding** $sgn\text{-}real\text{-}def\ zero\text{-}def$
 $one\text{-}def\ neg\text{-}one\text{-}def\ eudoxus\text{-}sgn\text{-}def$ **by** $simp$

lemma $sgn\text{-}abs\text{-}real\text{-}zero\text{-}iff$:
assumes $slope\ f$
shows $sgn\ (abs\text{-}real\ f) = 0 \longleftrightarrow (eudoxus\text{-}sgn\ f = (\lambda\cdot. 0))$ (**is** $?lhs \longleftrightarrow ?rhs$)
using $eudoxus\text{-}sgn\text{-}cong[OF\ rep\text{-}real\text{-}abs\text{-}real\text{-}refl,\ OF\ assms]$ $abs\text{-}real\text{-}eqI\ eudoxus\text{-}sgn\text{-}def$
 $neg\text{-}one\text{-}def\ one\text{-}def\ zero\text{-}def$
by $(auto\ simp\ add:\ sgn\text{-}real\text{-}def)$

lemma $sgn\text{-}zero\text{-}iff[simp]$: $sgn\ (x :: real) = 0 \longleftrightarrow x = 0$
using $eudoxus\text{-}sgn\text{-}iff(1)\ sgn\text{-}abs\text{-}real\text{-}zero\text{-}iff\ zero\text{-}iff\text{-}bounded'\ slope\text{-}refl$
by $(induct\ x)\ (metis\ (mono\text{-}tags)\ rep\text{-}real\text{-}abs\text{-}real\text{-}refl\ rep\text{-}real\text{-}iff)$

lemma $sgn\text{-}zero[simp]$: $sgn\ (0 :: real) = 0$ **by** $simp$

lemma $sgn\text{-}abs\text{-}real\text{-}one\text{-}iff$:
assumes $slope\ f$
shows $sgn\ (abs\text{-}real\ f) = 1 \longleftrightarrow pos\ f$
using $eudoxus\text{-}sgn\text{-}cong[OF\ rep\text{-}real\text{-}abs\text{-}real\text{-}refl,\ OF\ assms]$ $abs\text{-}real\text{-}eqI\ eudoxus\text{-}sgn\text{-}def$
 $neg\text{-}one\text{-}def\ one\text{-}def\ zero\text{-}def$
by $(auto\ simp\ add:\ sgn\text{-}real\text{-}def)$

lemmas $sgn\text{-}pos = sgn\text{-}abs\text{-}real\text{-}one\text{-}iff[THEN\ iffD2,\ simp]$

lemma $sgn\text{-}one[simp]$: $sgn\ (1 :: real) = 1$ **by** $(subst\ one\text{-}def)\ (fastforce\ simp\ add:\ pos\text{-}def\ iff:\ sgn\text{-}abs\text{-}real\text{-}one\text{-}iff)$

lemma $sgn\text{-}abs\text{-}real\text{-}neg\text{-}one\text{-}iff$:
assumes $slope\ f$
shows $sgn\ (abs\text{-}real\ f) = -\ 1 \longleftrightarrow neg\ f$

using *eudoxus-sgn-cong*[*OF rep-real-abs-real-refl, OF assms*] *abs-real-eqI eudoxus-sgn-def neg-one-def one-def zero-def pos-neg-exclusive*

by (*auto simp add: sgn-real-def*)

lemmas *sgn-neg = sgn-abs-real-neg-one-iff*[*THEN iffD2, simp*]

lemma *sgn-neg-one*[*simp*]: *sgn (- 1 :: real) = - 1* **by** (*subst neg-one-def*) (*fastforce simp add: neg-def eudoxus-uminus-def iff: sgn-abs-real-neg-one-iff*)

lemma *sgn-plus*:

assumes *sgn x = (1 :: real) sgn y = 1*

shows *sgn (x + y) = 1*

proof -

have *pos: pos (rep-real x) pos (rep-real y)* **using** *assms sgn-abs-real-one-iff*[*OF slope-rep-real*] **by** *simp+*

{

fix *C :: int* **assume** *C-nonneg: C ≥ 0*

then obtain *N M* **where** $\forall n \geq N. \text{rep-real } x \ n \geq C \ \forall n \geq M. \text{rep-real } y \ n \geq C$

using *pos unfolding pos-def* **by** *presburger*

hence $\forall n \geq \max N \ M. (\text{rep-real } x +_e \text{rep-real } y) \ n \geq C$ **using** *C-nonneg unfolding eudoxus-plus-def* **by** *fastforce*

hence $\exists N. \forall n \geq N. (\text{rep-real } x +_e \text{rep-real } y) \ n \geq C$ **by** *blast*

}

thus *?thesis* **using** *pos-def* **by** (*simp add: eudoxus-plus-cong plus-real-def*)

qed

lemma *sgn-times*: *sgn ((x :: real) * y) = sgn x * sgn y*

proof (*cases x = 0 ∨ y = 0*)

case *False*

have $*$: $\llbracket x \neq 0; \text{pos} (\text{rep-real } y) \rrbracket \implies \text{sgn} ((x :: \text{real}) * y) = \text{sgn } x * \text{sgn } y$ **for** *x y*

proof (*induct x rule: slope-induct, induct y rule: slope-induct*)

case (*slope y x*)

hence *pos-y: pos y* **using** *pos-cong* **by** *blast*

show *?case*

proof (*cases pos x*)

case *pos-x: True*

{

fix *C :: int* **assume** *asm: C ≥ 0*

then obtain *N* **where** $N: \forall n \geq N. x \ n \geq C$ **using** *pos-x unfolding pos-def*

by *blast*

then obtain *N'* **where** $\forall n \geq N'. y \ n \geq \max 0 \ N$ **using** *pos-y unfolding pos-def* **by** (*meson max.cobounded1*)

hence $\exists N'. \forall n \geq N'. x (y \ n) \geq C$ **using** *N* **by** *force*

}

hence *pos (x *_e y)* **unfolding** *pos-def eudoxus-times-def* **by** *simp*

thus *?thesis* **using** *pos-x pos-y slope* **by** (*simp add: eudoxus-times-def*)

next

case *-: False*

hence $neg\text{-}x$: $neg\ x$ **using** $slope$ **by** ($metis\ abs\text{-}real\text{-}eqI\ neg\text{-}iff\text{-}nonpos\text{-}nonzero$
 $zero\text{-}def\ zero\text{-}iff\text{-}bounded$)

 {

 fix $C :: int$ **assume** $C \geq 0$

 then obtain N **where** $N: \forall n \geq N. x\ n \leq -\ C$ **using** $neg\text{-}x$ **unfolding**
 $neg\text{-}def$ **by** $blast$

 then obtain N' **where** $\forall n \geq N'. y\ n \geq \max\ 0\ N$ **using** $pos\text{-}y$ **unfolding**
 $pos\text{-}def$ **by** ($meson\ max.\text{cobounded}1$)

 hence $\exists N'. \forall n \geq N'. x\ (y\ n) \leq -\ C$ **using** N **by** $force$

 }

hence $neg\ (x *_e y)$ **unfolding** $neg\text{-}def\ eudoxus\text{-}times\text{-}def$ **by** $simp$

thus $?thesis$ **using** $neg\text{-}x\ pos\text{-}y\ slope$ **by** ($simp\ add:\ eudoxus\text{-}times\text{-}def$)

qed

qed

moreover have $sgn\ ((x :: real) * y) = sgn\ x * sgn\ y$ **if** $neg\text{-}x$: $neg\ (rep\text{-}real\ x)$

and $neg\text{-}y$: $neg\ (rep\text{-}real\ y)$ **for** $x\ y$

proof –

 have $pos\text{-}uminus\text{-}y$: $pos\ (rep\text{-}real\ (-\ y))$ **by** ($metis\ abs\text{-}real\text{-}eq\text{-}iff\ eudoxus\text{-}uminus\text{-}cong$
 $map\text{-}fun\text{-}apply\ neg\text{-}iff\text{-}pos\text{-}uminus\ neg\text{-}y\ pos\text{-}cong\ rep\text{-}real\text{-}abs\text{-}real\text{-}refl\ rep\text{-}real\text{-}iff$
 $uminus\text{-}real\text{-}def$)

 moreover have $x \neq 0$ **using** $neg\text{-}iff\text{-}nonpos\text{-}nonzero\ neg\text{-}x\ zero\text{-}iff\text{-}bounded'$ **by**
 $fastforce$

 ultimately have $sgn\ (-\ (x * y)) = -\ 1$ **using** $sgn\text{-}neg[OF\ slope\text{-}rep\text{-}real\ neg\text{-}x]$
 $sgn\text{-}pos[OF\ slope\text{-}rep\text{-}real\ pos\text{-}uminus\text{-}y]$ **by** $fastforce$

 hence $pos\ (rep\text{-}real\ (x * y))$ **by** ($metis\ eudoxus\text{-}uminus\text{-}cong\ map\text{-}fun\text{-}apply$
 $pos\text{-}iff\text{-}neg\text{-}uminus\ sgn\text{-}abs\text{-}real\text{-}neg\text{-}one\text{-}iff\ slope\text{-}refl\ slope\text{-}rep\text{-}real\ uminus\text{-}real\text{-}def$)

 thus $?thesis$ **using** $sgn\text{-}neg[OF\ slope\text{-}rep\text{-}real]$ $sgn\text{-}pos[OF\ slope\text{-}rep\text{-}real]$ $neg\text{-}x$
 $neg\text{-}y$ **by** $simp$

 qed

 ultimately show $?thesis$ **using** $False\ neg\text{-}iff\text{-}nonpos\text{-}nonzero[OF\ slope\text{-}rep\text{-}real]$
 $zero\text{-}iff\text{-}bounded'$

 by ($cases\ pos\ (rep\text{-}real\ x)$; $cases\ pos\ (rep\text{-}real\ y)$) ($fastforce\ simp\ add:\ mult.\text{commute}$)+

qed ($force$)

lemma $sgn\text{-}uminus$: $sgn\ (-\ (x :: real)) = -\ sgn\ x$ **by** ($metis\ (mono\text{-}tags,\ lifting)$
 $mult\text{-}minus1\ sgn\text{-}neg\text{-}one\ sgn\text{-}times$)

lemma $sgn\text{-}plus'$:

assumes $sgn\ x = (-1 :: real)$ $sgn\ y = -1$

shows $sgn\ (x + y) = -1$

using $assms\ sgn\text{-}uminus[of\ x]\ sgn\text{-}uminus[of\ y]\ sgn\text{-}uminus[of\ x + y]\ sgn\text{-}plus[of$
 $-\ x - y]$

by ($simp\ add:\ equation\text{-}minus\text{-}iff$)

lemma $pos\text{-}dual\text{-}def$:

assumes $slope\ f$

shows $pos\ f = (\forall C \geq 0. \exists N. \forall n \leq N. f\ n \leq -\ C)$

proof –

have $pos\ f = neg\ (f *_e (-_e\ id))$ **by** ($metis\ abs\text{-}real\text{-}eq\text{-}iff\ abs\text{-}real\text{-}times\ add.\text{inverse}\text{-}inverse$)

assms eudoxus-times-commute mult-minus1-right neg-one-def sgn-abs-real-neg-one-iff sgn-abs-real-one-iff sgn-uminus slope-neg-one

also have ... = $(\forall C \geq 0. \exists N. \forall n \geq N. (f (- n)) \leq -C)$ **unfolding** *neg-def eudoxus-times-def eudoxus-uminus-def* **by** *simp*

also have ... = $(\forall C \geq 0. \exists N. \forall n \leq N. f n \leq -C)$ **by** (*metis add.inverse-inverse minus-le-iff*)

finally show *?thesis* .

qed

lemma *neg-dual-def*:

assumes *slope f*

shows $neg f = (\forall C \geq 0. \exists N. \forall n \leq N. f n \geq C)$

unfolding *neg-iff-pos-uminus* **using** *assms* **by** (*subst pos-dual-def*) (*auto simp add: eudoxus-uminus-def*)

lemma *pos-representative*:

assumes *slope f pos f*

obtains *g* **where** $f \sim_e g \wedge n. n \geq N \implies g n \geq C$

proof –

obtain N' **where** $N': \forall z \geq N'. f z \geq \max 0 C$ **using** *assms* **unfolding** *pos-def* **by** (*meson max.cobounded1*)

have *: $1 = abs-real (\lambda x. x + N' - N)$ *slope* $(\lambda x. x + N' - N)$ **unfolding** *one-def* **by** (*intro abs-real-eqI*) (*auto simp add: eudoxus-rel-def slope-def intro!: boundedI*)

hence $abs-real f * 1 = abs-real (f *_e (\lambda x. x + N' - N))$ **using** *abs-real-times[OF assms(1) *(2)]* **by** *simp*

hence $f \sim_e (f *_e (\lambda x. x + N' - N))$ **using** *assms ** **by** (*metis abs-real-eq-iff eudoxus-times-commute mult.right-neutral*)

moreover have $\forall z \geq N. (f *_e (\lambda x. x + N' - N)) z \geq C$ **unfolding** *eudoxus-times-def* **using** N' **by** *simp*

ultimately show *?thesis* **using** *that* **by** *blast*

qed

lemma *pos-representative'*:

assumes *slope f pos f*

obtains *g* **where** $f \sim_e g \wedge n. g n \geq C \implies n \geq N$

proof –

obtain N' **where** $\forall z \leq N'. f z \leq -(\max 0 (- C) + 1)$ **using** *assms* **unfolding** *pos-dual-def[OF assms(1)]* **by** (*metis max.cobounded1 add-increasing2 zero-less-one-class.zero-le-one*)

hence $N': \forall z \leq N'. f z < \min 0 C$ **by** *fastforce*

have *: $1 = abs-real (\lambda x. x + N' - N)$ *slope* $(\lambda x. x + N' - N)$ **unfolding** *one-def* **by** (*intro abs-real-eqI*) (*auto simp add: eudoxus-rel-def slope-def intro!: boundedI*)

hence $abs-real f * 1 = abs-real (f *_e (\lambda x. x + N' - N))$ **using** *abs-real-times[OF assms(1) *(2)]* **by** *simp*

hence $f \sim_e (f *_e (\lambda x. x + N' - N))$ **using** *assms ** **by** (*metis abs-real-eq-iff eudoxus-times-commute mult.right-neutral*)

moreover have $\forall z < N. (f *_e (\lambda x. x + N' - N)) z < C$ **unfolding** *eu-*

doxus-times-def using N' by *simp*
ultimately show *?thesis* using that by (*meson linorder-not-less*)
qed

lemma *neg-representative*:

assumes *slope f neg f*
obtains g where $f \sim_e g \wedge n. n \geq N \implies g n \leq -C$
proof –
obtain N' where $\forall z \geq N'. f z \leq -\max 0 C$ using *assms unfolding neg-def* by (*meson max.cobounded1*)
hence N' : $\forall z \geq N'. f z \leq \min 0 (-C)$ by *force*
have $*$: $1 = \text{abs-real } (\lambda x. x + N' - N)$ *slope* $(\lambda x. x + N' - N)$ **unfolding** *one-def* by (*intro abs-real-eqI*) (*auto simp add: eudoxus-rel-def slope-def intro! boundedI*)
hence $\text{abs-real } f * 1 = \text{abs-real } (f *_e (\lambda x. x + N' - N))$ using *abs-real-times[OF assms(1) *(2)]* by *simp*
hence $f \sim_e (f *_e (\lambda x. x + N' - N))$ using *assms ** by (*metis abs-real-eq-iff eudoxus-times-commute mult.right-neutral*)
moreover **have** $\forall z \geq N. (f *_e (\lambda x. x + N' - N)) z \leq -C$ **unfolding** *eudoxus-times-def* using N' by *simp*
ultimately show *?thesis* using that by *blast*
qed

lemma *neg-representative'*:

assumes *slope f neg f*
obtains g where $f \sim_e g \wedge n. g n \leq -C \implies n \geq N$
proof –
obtain N' where $\forall z \leq N'. f z \geq \max 0 (-C) + 1$ using *assms unfolding neg-dual-def[OF assms(1)]* by (*metis max.cobounded1 add-increasing2 zero-less-one-class.zero-le-one*)
hence N' : $\forall z \leq N'. f z > \max 0 (-C)$ by *fastforce*
have $*$: $1 = \text{abs-real } (\lambda x. x + N' - N)$ *slope* $(\lambda x. x + N' - N)$ **unfolding** *one-def* by (*intro abs-real-eqI*) (*auto simp add: eudoxus-rel-def slope-def intro! boundedI*)
hence $\text{abs-real } f * 1 = \text{abs-real } (f *_e (\lambda x. x + N' - N))$ using *abs-real-times[OF assms(1) *(2)]* by *simp*
hence $f \sim_e (f *_e (\lambda x. x + N' - N))$ using *assms ** by (*metis abs-real-eq-iff eudoxus-times-commute mult.right-neutral*)
moreover **have** $\forall z < N. (f *_e (\lambda x. x + N' - N)) z > -C$ **unfolding** *eudoxus-times-def* using N' by *simp*
ultimately show *?thesis* using that by (*meson linorder-not-less*)
qed

We call a real x less than another real y , if their difference is positive.

definition

$$x < (y::\text{real}) \equiv \text{sgn } (y - x) = 1$$

definition

$$x \leq (y::\text{real}) \equiv x < y \vee x = y$$

definition

abs-real: $|x :: \text{real}| = (\text{if } 0 \leq x \text{ then } x \text{ else } -x)$

instance ..

end

instance *real* :: *linorder*

proof

fix $x\ y\ z :: \text{real}$

show $(x < y) = (x \leq y \wedge \neg y \leq x)$ **unfolding** *less-eq-real-def less-real-def* **using** *sgn-times[of -1 x - y]* **by** *fastforce*

show $x \leq x$ **unfolding** *less-eq-real-def* **by** *blast*

show $\llbracket x \leq y; y \leq z \rrbracket \implies x \leq z$ **unfolding** *less-eq-real-def less-real-def* **using** *sgn-plus* **by** *fastforce*

show $\llbracket x \leq y; y \leq x \rrbracket \implies x = y$ **unfolding** *less-eq-real-def less-real-def* **using** *sgn-times[of -1 x - y]* **by** *fastforce*

show $x \leq y \vee y \leq x$ **unfolding** *less-eq-real-def less-real-def* **using** *sgn-times[of -1 x - y]* *sgn-range* **by** *force*

qed

lemma *real-leI*:

assumes $\text{sgn } (y - x) \in \{0 :: \text{real}, 1\}$

shows $x \leq y$

using *assms* **unfolding** *less-eq-real-def less-real-def* **by** *force*

lemma *real-lessI*:

assumes $\text{sgn } (y - x) = (1 :: \text{real})$

shows $x < y$

using *assms* **unfolding** *less-real-def* **by** *blast*

lemma *abs-real-leI*:

assumes $\text{slope } f \text{ slope } g \wedge z. z \geq N \implies f z \geq g z$

shows $\text{abs-real } f \geq \text{abs-real } g$

proof –

{

assume $\text{abs-real } f \neq \text{abs-real } g$

hence $\text{abs-real } (f +_e -_e g) \neq 0$ **by** (*metis abs-real-minus assms(1,2) eq-iff-diff-eq-0*)

hence $\neg \text{bounded } (f +_e -_e g)$ **by** (*metis abs-real-eqI zero-def zero-iff-bounded*)

hence $\text{pos } (f +_e -_e g) \vee \text{neg } (f +_e -_e g)$ **using** *assms eudoxus-plus-cong eudoxus-uminus-cong neg-iff-nonpos-nonzero slope-refl* **by** *auto*

moreover

{

assume $\text{neg } (f +_e -_e g)$

then obtain N' **where** $(f +_e -_e g) z \leq -1$ **if** $z \geq N'$ **for** z **unfolding** *neg-def* **by** *fastforce*

hence $f z < g z$ **if** $z \geq N'$ **for** z **using** *that* **unfolding** *eudoxus-plus-def eudoxus-uminus-def* **by** *fastforce*

hence *False* **using** *assms* **by** (*metis linorder-not-less nle-le*)

```

    }
    ultimately have abs-real f > abs-real g using assms by (fastforce intro:
real-lessI sgn-pos simp add: eudoxus-plus-def eudoxus-uminus-def)
  }
  thus ?thesis unfolding less-eq-real-def by argo
qed

```

lemma *abs-real-lessI*:

```

  assumes slope f slope g  $\wedge z. z \geq N \implies f z \geq g z \wedge C. C \geq 0 \implies \exists z. f z \geq g z + C$ 
  shows abs-real f > abs-real g
proof -
  {
    assume bounded (f +e -e g)
    then obtain C where  $|f z - g z| \leq C \wedge C \geq 0$  for z unfolding eudoxus-plus-def
eudoxus-uminus-def by auto
    moreover obtain z where  $f z \geq g z + (C + 1)$  using assms(4)[of C + 1]
    calculation by auto
    ultimately have False by (metis abs-le-D1 add.commute dual-order.trans
le-diff-eq linorder-not-less zless-add1-eq)
  }
  moreover have abs-real f ≥ abs-real g using assms abs-real-leI by blast
  ultimately show ?thesis by (metis abs-real-minus assms(1,2) eq-iff-diff-eq-0
eudoxus-plus-cong eudoxus-sgn-iff(1) eudoxus-uminus-cong order-le-imp-less-or-eq
sgn-abs-real-zero-iff sgn-zero slope-refl)
qed

```

lemma *abs-real-lessD*:

```

  assumes slope f slope g abs-real f > abs-real g
  obtains z where  $z \geq N \wedge f z > g z$ 
proof -
  {
    assume  $\exists N. \forall z \geq N. f z \leq g z$ 
    then obtain N where  $f z \leq g z$  if  $z \geq N$  for z by fastforce
    hence False using assms abs-real-leI by (metis linorder-not-le)
  }
  thus ?thesis using that by fastforce
qed

```

2.5 Multiplicative Inverse

We now define the multiplicative inverse. We start by constructing a candidate for positive slopes first and then extend it to the entire domain using the choice function *Eps*.

```

instantiation real :: {inverse}
begin

```

```

definition eudoxus-pos-inverse ::  $(int \Rightarrow int) \Rightarrow (int \Rightarrow int)$  where
  eudoxus-pos-inverse f z = sgn z * Inf ({0..}  $\cap$  {n. f n ≥ |z|})

```

lemma *eudoxus-pos-inverse*:

assumes *slope f pos f*

obtains *g* **where** $f \sim_e g$ *slope (eudoxus-pos-inverse g) eudoxus-pos-inverse g *_e f ~_e id*

proof –

let $?φ = \text{eudoxus-pos-inverse}$

obtain *g* **where** $g: f \sim_e g$ $g z \geq 0 \implies z > 1$ **for** *z* **using** *pos-representative'[OF assms]* **by** (*metis gt-ex order-less-le-trans*)

hence *pos-g: pos g* **using** *assms pos-cong* **by** *blast*

have *slope-g: slope g* **using** *g unfolding eudoxus-rel-def* **by** *simp*

have $\exists n \geq 0. g n \geq |z|$ **for** *z* **using** *pos-g unfolding pos-def* **by** (*metis abs-ge-self order-less-imp-le zero-less-abs-iff*)

hence *nonempty-φ: {0..} ∩ {n. |z| ≤ g n} ≠ {}* **for** *z* **by** *blast*

have *bdd-below-φ: bdd-below ({0..} ∩ {n. g n ≥ |z|})* **for** *z* **by** *simp*

have *φ-bound: g n ≥ z ⟹ ?φ g z ≤ n* **if** $z \geq 0$ $n \geq 0$ **for** *n z* **unfolding** *eudoxus-pos-inverse-def* **using** *cInf-lower[OF - bdd-below-φ, of n z]* **that** *abs-of-nonneg zsgn-def* **by** *simp*

hence *φ-bound'*: $?φ g z > n \implies g n < z$ **if** $z \geq 0$ $n \geq 0$ **for** *z n* **using** *that linorder-not-less* **by** *blast*

have *φ-mem: z > 0 ⟹ ?φ g z ∈ {0..} ∩ {n. g n ≥ |z|}* **for** *z* **unfolding** *eudoxus-pos-inverse-def* **using** *int-Inf-mem[OF nonempty-φ bdd-below-φ, of z]* **by** *simp*

obtain *L* **where** $|g(1 + (z - 1)) - (g 1 + g(z - 1))| \leq L$ **for** *z* **using** *slope-g* **by** *fast*

hence $*$: $|g z - (g 1 + g(z - 1))| \leq L$ **for** *z* **by** *simp*

hence *L: g z ≤ g(z - 1) + (L + g 1)* **for** *z* **using** *abs-le-D1 *[of z]* **by** *linarith*

let $?γ = λ m n. (g(m + (-n)) - (g m + g(-n))) - (g(n + (-n)) - (g n + g(-n))) + g 0$

obtain *c* **where** $c: |g(m + (-n)) - (g m + g(-n))| \leq c$ **for** *m n* **using** *slope-g* **by** *fast*

obtain *c'* **where** $c': |g(n + (-n)) - (g n + g(-n))| \leq c'$ **for** *n* **using** *slope-g* **by** *fast*

have $|?γ m n| \leq |g(m + (-n)) - (g m + g(-n))| + |g(n + (-n)) - (g n + g(-n))| + |g 0|$ **for** *m n* **by** *linarith*

hence $*$: $|?γ m n| \leq c + c' + |g 0|$ **for** *m n* **using** *c[of m n] c'[of n]* **by** *linarith*

define *C* **where** $C = 2 * (c + c' + |g 0|)$

have $g(m - (n + p)) - (g m - (g n + g p)) = ?γ(m - n) p + ?γ m n$ **for** *m n p* **by** (*simp add: algebra-simps*)

hence $|g(m - (n + p)) - (g m - (g n + g p))| \leq (c + c' + |g 0|) + (c + c' + |g 0|)$ **for** *m n p* **using** $*$ [*of m - n p*] $*$ [*of m n*] **by** *simp*

hence $*$: $|g(m - (n + p)) - (g m - (g n + g p))| \leq C$ **for** *m n p* **unfolding** *C-def* **by** (*metis mult-2*)

have $C: g(m - (n + p)) \leq g m - (g n + g p) + C g m - (g n + g p) + (- C)$
 $\leq g(m - (n + p))$ **for** $m n p$ **using** $*[of m n p]$ *abs-le-D1 abs-le-D2* **by** *linarith+*

have *bounded: bounded h* **if** *bounded: bounded (g o h)* **for** $h :: 'a \Rightarrow int$
proof (*rule ccontr*)
assume *asm: \neg bounded h*
obtain C **where** $C: |g(h z)| \leq C C \geq 0$ **for** z **using** *bounded* **by** *fastforce*
obtain N **where** $N: g z \geq C + 1$ **if** $z \geq N$ **for** z **using** C *pos-g unfolding*
pos-def **by** *fastforce*
obtain N' **where** $N': g z \leq -(C + 1)$ **if** $z \leq N'$ **for** z **using** C *pos-g unfolding*
pos-dual-def[OF slope-g] **by** (*meson add-increasing2 linordered-nonzero-semiring-class.zero-le-one*)
obtain z **where** $|h z| > \max |N| |N'|$ **using** *asm unfolding bounded-alt-def*
by (*meson leI*)
hence $h z \in \{..N'\} \cup \{N..\}$ **by** *fastforce*
hence $g(h z) \in \{..-(C + 1)\} \cup \{C + 1..\}$ **using** $N N'$ **by** *blast*
hence $|g(h z)| \geq C + 1$ **by** *fastforce*
thus *False* **using** $C(1)[of z]$ **by** *simp*
qed

define D **where** $D = \max |-(C + (L + g 1) + (L + g 1))| |C + L + g 1|$
 $\{$
fix $m n :: int$
assume *asm: $m > 0 n > 0$*

have $g(? \varphi g m) \geq m$ **using** *φ -mem asm* **by** *simp*
moreover **have** $? \varphi g m > 1$ **using** *calculation g asm* **by** *simp*
moreover **have** $m > g(? \varphi g m - 1)$ **using** *asm calculation* **by** (*intro φ -bound'*)
auto
ultimately **have** $m: m \in \{g(? \varphi g m - 1) < .. g(? \varphi g m)\}$ **by** *simp*

have $g(? \varphi g n) \geq n$ **using** *φ -mem asm* **by** *simp*
moreover **have** $? \varphi g n > 1$ **using** *calculation g asm* **by** *simp*
moreover **have** $n > g(? \varphi g n - 1)$ **using** *asm calculation* **by** (*intro φ -bound'*)
auto
ultimately **have** $n: n \in \{g(? \varphi g n - 1) < .. g(? \varphi g n)\}$ **by** *simp*

have $g(? \varphi g (m + n)) \geq m + n$ **using** *φ -mem asm* **by** *simp*
moreover **have** $? \varphi g (m + n) > 1$ **using** *calculation g asm* **by** *simp*
moreover **have** $(m + n) > g(? \varphi g (m + n) - 1)$ **using** *asm calculation* **by**
(*intro φ -bound'*) *auto*
ultimately **have** $m+n: m + n \in \{g(? \varphi g (m + n) - 1) < .. g(? \varphi g (m + n))\}$
by *simp*

have $*$: $g(? \varphi g (m + n)) - (g(? \varphi g m - 1) + g(? \varphi g n - 1)) > 0$ $g(? \varphi g (m + n) - 1) - (g(? \varphi g m) + g(? \varphi g n)) < 0$ **using** $m-n m n$ **by** *simp+*

have $g(? \varphi g (m + n) - (? \varphi g m + ? \varphi g n)) \leq g(? \varphi g (m + n)) - (g(? \varphi g m) + g(? \varphi g n)) + C$ **using** C **by** *blast*
also **have** $... \leq g(? \varphi g (m + n) - 1) - g(? \varphi g m) - g(? \varphi g n) + (C + L$

+ $g\ 1$) **using** L **by** *fastforce*
finally have *upper*: $g\ (?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n)) ≤ C + L + g\ 1$
using $*$ **by** *fastforce*

have $-(C + (L + g\ 1) + (L + g\ 1)) ≤ g\ (?φ\ g\ (m + n)) - g\ (?φ\ g\ m - 1)$
 $- g\ (?φ\ g\ n - 1) - (C + (L + g\ 1) + (L + g\ 1))$ **using** $*$ **by** *linarith*
also have $... ≤ g\ (?φ\ g\ (m + n)) - (g\ (?φ\ g\ m) + g\ (?φ\ g\ n)) + (-C)$ **using**
 $L[THEN\ le-imp-neg-le,\ of\ ?φ\ g\ m]\ L[THEN\ le-imp-neg-le,\ of\ ?φ\ g\ n]$ **by** *linarith*
also have $... ≤ g\ (?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n))$ **using** C **by** *blast*
finally have *lower*: $-(C + (L + g\ 1) + (L + g\ 1)) ≤ g\ (?φ\ g\ (m + n) -$
 $(?φ\ g\ m + ?φ\ g\ n))$.

have $|g\ (?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n))| ≤ D$ **using** *upper lower*
unfolding D-def **by** *simp*

}
hence *bounded* $(g\ o\ (λ(m, n).\ ?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n))\ o\ (λ(m, n).$
 $(max\ 1\ m,\ max\ 1\ n)))$ **by** *(intro boundedI[of - D]) auto*

hence *bounded* $((λ(m, n).\ ?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n))\ o\ (λ(m, n).$
 $(max\ 1\ m,\ max\ 1\ n)))$ **by** *(metis (mono-tags, lifting) bounded comp-assoc)*

then obtain C **where** $|((λ(m, n).\ ?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n))\ o\ (λ(m,$
 $n). (max\ 1\ m,\ max\ 1\ n)))\ (m, n)| ≤ C$ **for** $m\ n$ **by** *blast*

hence $|?φ\ g\ (m + n) - (?φ\ g\ m + ?φ\ g\ n)| ≤ C$ **if** $m ≥ 1\ n ≥ 1$ **for** $m\ n$ **using**
that[THEN max-absorb2] **by** *(metis (no-types, lifting) comp-apply prod.case)*

hence *slope*: *slope* $(?φ\ g)$ **by** *(intro slope-odd[of - C]) (auto simp add: eu-*
dorus-pos-inverse-def)

moreover

{
obtain C **where** $C: |g\ ((?φ\ g\ n - 1) + 1) - (g\ (?φ\ g\ n - 1) + g\ 1)| ≤ C$
for n **using** *slope-g* **by** *fast*

have C -*bound*: $g\ (?φ\ g\ n - 1) ≥ g\ (?φ\ g\ n) - (|g\ 1| + C)$ **for** n **using** C [*of*
 $n]$ **by** *fastforce*

{
fix $n :: int$
assume *asm*: $n > 0$
have *upper*: $g\ (?φ\ g\ n) ≥ n$ **using** *φ-mem asm* **by** *simp*
moreover have $?φ\ g\ n > 1$ **using** *calculation g asm* **by** *simp*
moreover have $n > g\ (?φ\ g\ n - 1)$ **using** *calculation asm* **by** *(intro φ-bound')*

auto

moreover have $n ≥ g\ (?φ\ g\ n) - (|g\ 1| + C)$ **using** *calculation C-bound[of*
 $n]$ **by** *force*

ultimately have $|g\ (?φ\ g\ n) - n| ≤ |g\ 1| + C$ **by** *simp*

}
hence *id*: $g\ *_e\ ?φ\ g\ \sim_e\ id$ **using** *slope-g slope* **by** *(intro eudorus-rell[of - - 1*
 $|g\ 1| + C]) (auto simp add: eudorus-times-def)$

}
ultimately show *?thesis* **using** g **that** *eudorus-rel-trans eudorus-times-cong*
slope-refl eudorus-times-commute[OF slope slope-g] **by** *metis*

qed

definition *eudoxus-inverse* :: (int \Rightarrow int) \Rightarrow (int \Rightarrow int) **where**
eudoxus-inverse f = (if \neg bounded f then SOME g. slope g \wedge (g *_e f) \sim_e id else
(λ -. 0))

lemma

assumes slope f

shows slope-eudoxus-inverse: slope (eudoxus-inverse f) (is ?slope) **and**

eudoxus-inverse-id: \neg bounded f \Longrightarrow eudoxus-inverse f *_e f \sim_e id (is \neg
bounded f \Longrightarrow ?id)

proof –

have *: \llbracket slope g; (g *_e f) \sim_e id $\rrbracket \Longrightarrow$?slope \llbracket slope g; (g *_e f) \sim_e id; \neg bounded
f $\rrbracket \Longrightarrow$?id **for** g

unfolding eudoxus-inverse-def **using** someI[**where** ?P= λ g. slope g \wedge (g *_e f)
 \sim_e id] **by** auto

{

assume pos: pos f

then obtain g **where** slope (eudoxus-pos-inverse g) eudoxus-pos-inverse g *_e
f \sim_e id **using** eudoxus-pos-inverse[OF assms] **by** blast

hence ?slope \neg bounded f \Longrightarrow ?id **using** pos pos-iff-nonneg-nonnzero[OF assms]
* **by** blast+

}

moreover

{

assume nonpos: \neg pos f

{

assume nonzero: \neg bounded f

hence uminus-f: slope ($-_e$ f) pos ($-_e$ f) **using** neg-iff-pos-uminus neg-iff-nonpos-nonnzero
assms slope-refl nonpos **by** auto

then obtain g **where** g: slope (eudoxus-pos-inverse g) eudoxus-pos-inverse g
*_e ($-_e$ f) \sim_e id **using** eudoxus-pos-inverse **by**metis

hence $-_e$ (eudoxus-pos-inverse g) *_e f \sim_e id **by** (metis (full-types) uminus-f(1)
abs-real-eq-iff abs-real-times abs-real-uminus assms(1) eudoxus-times-commute mi-
nus-mult-commute rel-funE uminus-real.rsp)

moreover have slope ($-_e$ (eudoxus-pos-inverse g)) **using** uminus-f eu-
doxus-uminus-cong slope-refl g **by** presburger

ultimately have ?slope ?id **using** * nonzero **by** blast+

}

moreover have bounded f \Longrightarrow ?slope **unfolding** eudoxus-inverse-def **by** simp

ultimately have ?slope \neg bounded f \Longrightarrow ?id **by** blast+

}

ultimately show ?slope \neg bounded f \Longrightarrow ?id **by** blast+

qed

quotient-definition

(inverse :: real \Rightarrow real) **is** eudoxus-inverse

proof –

fix x x' **assume** asm: x \sim_e x'

hence slopes: slope x slope x' **unfolding** eudoxus-rel-def **by** blast+

show eudoxus-inverse x \sim_e eudoxus-inverse x'

```

proof (cases bounded x)
  case True
    hence bounded x' by (meson asm eudoxus-rel-sym eudoxus-rel-trans zero-iff-bounded)
    then show ?thesis unfolding eudoxus-inverse-def using True slope-zero
slope-refl by auto
  next
    case False
      hence ¬ bounded x' by (meson asm eudoxus-rel-sym eudoxus-rel-trans zero-iff-bounded)
      hence inverses: eudoxus-inverse x *e x ~e id eudoxus-inverse x' *e x' ~e id
using slopes eudoxus-inverse-id False by blast+

    have alt-inverse: eudoxus-inverse x *e x' ~e id
      using inverses eudoxus-times-cong[OF slope-reflI, OF slope-eudoxus-inverse
asm, OF slopes(1)]
      eudoxus-rel-sym eudoxus-rel-trans by blast

    have eudoxus-inverse x ~e eudoxus-inverse x *e (eudoxus-inverse x' *e x')
      using eudoxus-times-cong[OF slope-reflI, OF slope-eudoxus-inverse inverses(2)][THEN
eudoxus-rel-sym], OF slopes(1)]
      by (simp add: eudoxus-times-def)
    also have ... ~e eudoxus-inverse x' *e (eudoxus-inverse x *e x')
      using eudoxus-times-commute[OF slope-eudoxus-inverse(1,1), OF slopes,
THEN eudoxus-times-cong, OF slope-reflI, OF slopes(2)]
      by (simp add: eudoxus-times-def comp-assoc)
    also have ... ~e eudoxus-inverse x' *e id using alt-inverse eudoxus-times-cong[OF
slope-reflI] slope-eudoxus-inverse slopes by blast
    also have ... = eudoxus-inverse x' unfolding eudoxus-times-def by simp
    finally show ?thesis .
  qed
qed

```

definition

$$x \text{ div } (y::\text{real}) = \text{inverse } y * x$$

instance ..

end

lemmas eudoxus-inverse-cong = apply-rsp'[OF inverse-real.rsp, intro]

lemma eudoxus-inverse-abs[simp]:

assumes slope f ¬ bounded f

shows inverse (abs-real f) * abs-real f = 1

unfolding inverse-real-def **using** eudoxus-inverse-id[OF assms]

by (metis abs-real-eqI abs-real-times assms(1) eudoxus-inverse-cong map-fun-apply
one-def rep-real-abs-real-refl slope-refl)

The Eudoxus reals are a field, with inverses defined as above.

instance real :: field

proof

```

fix x y :: real
show x ≠ 0 ⇒ inverse x * x = 1 using eudoxus-sgn-iff(1) sgn-abs-real-zero-iff
by (induct x rule: slope-induct) force
show x / y = x * inverse y unfolding divide-real-def by simp
show inverse (0 :: real) = 0 unfolding inverse-real-def eudoxus-inverse-def
using zero-def zero-iff-bounded' by auto
qed

```

```

instantiation real :: distrib-lattice
begin

```

```

definition
  (inf :: real ⇒ real ⇒ real) = min

```

```

definition
  (sup :: real ⇒ real ⇒ real) = max

```

```

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

```

```

end

```

The ordering on the Eudoxus reals is linear.

```

instance real :: linordered-field

```

```

proof

```

```

  fix x y z :: real
  show z + x ≤ z + y if x ≤ y
  proof (cases x = y)
    case False
      hence x < y using that by simp
      thus ?thesis
    proof (induct x rule: slope-induct, induct y rule: slope-induct, induct z rule: slope-induct)
      case (slope h g f)
        hence pos (g +e (-e f)) unfolding less-real-def using sgn-abs-real-one-iff
  by (force simp add: eudoxus-plus-def eudoxus-uminus-def)
    thus ?case by (metis slope(4) less-real-def add-diff-cancel-left nless-le)
  qed
qed (force)

```

```

show |x| = (if x < 0 then -x else x) by (metis abs-real less-eq-real-def not-less-iff-gr-or-eq)
show sgn x = (if x = 0 then 0 else if 0 < x then 1 else - 1) using sgn-range sgn-zero-iff by (auto simp: less-real-def)
show  $\llbracket x < y; 0 < z \rrbracket \implies z * x < z * y$  by (metis (no-types, lifting) diff-zero less-real-def mult.right-neutral right-diff-distrib' sgn-times)
qed

```

The Eudoxus reals fulfill the Archimedean property.

```

instance real :: archimedean-field

```

```

proof

```

```

fix  $x :: \text{real}$ 
show  $\exists z. x \leq \text{of-int } z$ 
proof (induct x rule: slope-induct)
  case (slope y)
    then obtain  $A B$  where linear-bound:  $|y z| \leq A * |z| + B$   $0 \leq A$   $0 \leq B$  for
     $z$  using slope-linear-bound by blast
    {
      fix  $C$  assume C-nonneg:  $0 \leq (C :: \text{int})$ 
      {
        fix  $z$  assume asm:  $z \geq B + C$ 
        have  $y z + C \leq A * |z| + B + C$  using abs-le-D1 linear-bound by auto
        also have  $\dots \leq (A + 1) * |z|$  using C-nonneg linear-bound(2,3) asm by
        (auto simp: distrib-right)
        finally have  $y z + C \leq (A + 1) * z$  using add-nonneg-nonneg[OF C-nonneg
        linear-bound(3)] abs-of-nonneg[of z] asm by linarith
      }
      hence  $\exists N. \forall x \geq N. (((*) (A + 1)) +_e -_e y) x \geq C$  unfolding eudoxus-plus-def eudoxus-uminus-def by fastforce
    }
    hence pos  $((*) (A + 1)) +_e -_e y$  unfolding pos-def by blast
    hence pos (rep-real (of-int  $(A + 1) - \text{abs-real } y$ )) unfolding real-of-int using
    slope by (simp, subst pos-cong[OF rep-real-abs-real-refl]) (auto simp add: eudoxus-plus-def eudoxus-uminus-def)
    hence abs-real  $y < \text{of-int } (A + 1)$  unfolding less-real-def by (metis sgn-pos
    rep-real-abs-real-refl rep-real-iff slope-rep-real)
    thus ?case unfolding less-eq-real-def by blast
  qed
qed

```

2.6 Completeness

To show that the Eudoxus reals are complete, we first introduce the floor function.

```

instantiation real :: floor-ceiling
begin

```

definition

```

(floor :: (real  $\Rightarrow$  int)) = ( $\lambda x. (\text{SOME } z. \text{of-int } z \leq x \wedge x < \text{of-int } z + 1)$ )

```

instance

proof

```

fix  $x :: \text{real}$ 
show of-int  $\lfloor x \rfloor \leq x \wedge x < \text{of-int } (\lfloor x \rfloor + 1)$  using someI[of  $\lambda z. \text{of-int } z \leq x \wedge$ 
 $x < \text{of-int } z + 1$ ] floor-exists by (fastforce simp add: floor-real-def)
qed
end

```

lemma *eudoxus-dense-rational*:

```

fixes  $x y :: \text{real}$ 

```

assumes $x < y$
obtains $m\ n$ **where** $x < (of\text{-}int\ m / of\text{-}int\ n) (of\text{-}int\ m / of\text{-}int\ n) < y\ n > 0$
proof –
obtain $n :: int$ **where** $n: inverse\ (y - x) < of\text{-}int\ n\ n > 0$ **by** (*metis ex-less-of-int antisym-conv3 dual-order.strict-trans of-int-less-iff*)
hence $*$: $inverse\ (of\text{-}int\ n) < y - x$ **by** (*metis assms diff-gt-0-iff-gt inverse-inverse-eq inverse-less-iff-less inverse-positive-iff-positive of-int-0-less-iff*)
define m **where** $m = floor\ (x * of\text{-}int\ n) + 1$
{
 assume $y \leq of\text{-}int\ m / of\text{-}int\ n$
 hence $inverse\ (of\text{-}int\ n) < of\text{-}int\ m / of\text{-}int\ n - x$ **using** $*$ **by** *linarith*
 hence $x < (of\text{-}int\ m - 1) / of\text{-}int\ n$ **by** (*simp add: diff-divide-distrib inverse-eq-divide*)
 hence *False* **unfolding** $m\text{-}def$ **using** $n(2)$ *divide-le-eq linorder-not-less* **by** *fastforce*
}
moreover **have** $x < of\text{-}int\ m / of\text{-}int\ n$ **unfolding** $m\text{-}def$ **by** (*meson n(2) floor-correct mult-imp-less-div-pos of-int-pos*)
ultimately show *?thesis* **using** *that n* **by** *fastforce*
qed

The Eudoxus reals are a complete field.

lemma *eudoxus-complete*:

assumes $S \neq \{\}$ *bdd-above* S
obtains $u :: real$ **where** $\bigwedge s. s \in S \implies s \leq u \wedge y. (\bigwedge s. s \in S \implies s \leq y) \implies u \leq y$

proof (*cases* $\exists u \in S. \forall s \in S. s \leq u$)

case *False*

hence *no-greatest-element*: $\exists y \in S. x < y$ **if** $x \in S$ **for** x **using** *that* **by** *force*

define $u :: int \Rightarrow int$ **where** $u = (\lambda z. sgn\ z * Sup\ ((\lambda x. \lfloor of\text{-}int\ |z| * x \rfloor) 'S))$

have *bdd-above-u*: *bdd-above* $((\lambda x. \lfloor of\text{-}int\ |z| * x \rfloor) 'S)$ **for** z **by** (*intro bdd-above-image-mono[OF - assms(2)] monoI*) (*simp add: floor-mono mult.commute mult-right-mono*)

have *u-Sup-nonneg*: $z \geq 0 \implies \lfloor of\text{-}int\ z * s \rfloor \leq u\ z$ **and**

u-Sup-nonpos: $z \leq 0 \implies - \lfloor of\text{-}int\ (-z) * s \rfloor \geq u\ z$ **if** $s \in S$ **for** $s\ z$

unfolding *u-def* **using** *cSup-upper[OF - bdd-above-u, of \lfloor of\text{-}int\ |z| * s \rfloor z]* *that abs-of-nonpos zsgn-def* **by** *force+*

have *u-mem*: $u\ z \in (\lambda x. sgn\ z * \lfloor of\text{-}int\ |z| * x \rfloor) 'S$ **for** z **unfolding** *u-def* **using** *int-Sup-mem[OF - bdd-above-u, of z] assms* **by** *auto*

have *slope*: *slope* u

proof –

{

fix $m\ n :: int$ **assume** *asm*: $m > 0\ n > 0$

obtain $x\text{-}m$ **where** $x\text{-}m: x\text{-}m \in S\ u\ m = \lfloor of\text{-}int\ m * x\text{-}m \rfloor$ **using** *u-mem[of m] asm zsgn-def* **by** *auto*

obtain $x\text{-}n$ **where** $x\text{-}n: x\text{-}n \in S\ u\ n = \lfloor of\text{-}int\ n * x\text{-}n \rfloor$ **using** *u-mem[of n]*

asm zsgn-def by auto

obtain $x\text{-}m\text{-}n$ **where** $x\text{-}m\text{-}n$: $x\text{-}m\text{-}n \in S$ $u(m+n) = \lfloor \text{of-int}(m+n) * x\text{-}m\text{-}n \rfloor$ **using** $u\text{-}mem[\text{of } m+n]$ *asm zsgn-def by auto*

define x **where** $x = \max(\max x\text{-}m x\text{-}n) x\text{-}m\text{-}n$
have $x: x \in S$ **unfolding** $x\text{-}def$ **using** $x\text{-}m x\text{-}n x\text{-}m\text{-}n$ **by** *linarith*

have $x \geq x\text{-}m$ $x \geq x\text{-}n$ $x \geq x\text{-}m\text{-}n$ **unfolding** $x\text{-}def$ **by** *linarith*+

hence $u m \leq \lfloor \text{of-int } m * x \rfloor$ $u n \leq \lfloor \text{of-int } n * x \rfloor$ $u(m+n) \leq \lfloor \text{of-int}(m+n) * x \rfloor$

unfolding $x\text{-}m x\text{-}n x\text{-}m\text{-}n$ **by** (*meson asm floor-less-cancel linorder-not-less mult-le-cancel-left-pos of-int-0-less-iff add-pos-pos*)+

hence $u m = \lfloor \text{of-int } m * x \rfloor$ $u n = \lfloor \text{of-int } n * x \rfloor$ $u(m+n) = \lfloor \text{of-int } m * x + \text{of-int } n * x \rfloor$

using $u\text{-}Sup\text{-}nonneg[OF x(1), of m]$ $u\text{-}Sup\text{-}nonneg[OF x(1), of n]$ $u\text{-}Sup\text{-}nonneg[OF x(1), of m+n]$ *asm add-pos-pos[OF asm]* **by** (*force simp add: distrib-right*)+

moreover

{

fix $a b :: \text{real}$

have $a - \text{of-int } \lfloor a \rfloor \in \{0..<1\}$ **using** *floor-less-one by fastforce*

moreover have $b - \text{of-int } \lfloor b \rfloor \in \{0..<1\}$ **using** *floor-less-one by fastforce*

ultimately have $(a - \text{of-int } \lfloor a \rfloor) + (b - \text{of-int } \lfloor b \rfloor) \in \{0..<2\}$ **unfolding**

atLeastLessThan-def by simp

hence $(a+b) - (\text{of-int } \lfloor a \rfloor + \text{of-int } \lfloor b \rfloor) \in \{0..<2\}$ **by** (*simp add: diff-add-eq*)

hence $\lfloor a+b - (\text{of-int } \lfloor a \rfloor + \text{of-int } \lfloor b \rfloor) \rfloor \in \{0..<2\}$ **by** *simp*

hence $\lfloor a+b \rfloor - (\lfloor a \rfloor + \lfloor b \rfloor) \in \{0..<2\}$ **by** (*metis floor-diff-of-int of-int-add*)

}

ultimately have $|u(m+n) - (u m + u n)| \leq 2$ **by** (*metis abs-of-nonneg atLeastLessThan-iff nless-le*)

}

moreover have $u z = - u(-z)$ **for** z **unfolding** $u\text{-}def$ **by** *simp*

ultimately show *?thesis* **using** *slope-odd by blast*

qed

{

fix s **assume** $s \in S$

then obtain y **where** $y: s < y$ $y \in S$ **using** *no-greatest-element by blast*

then obtain $m n :: \text{int}$ **where** $*$: $s < (\text{of-int } m / \text{of-int } n)$ $(\text{of-int } m / \text{of-int } n) < y$ $n > 0$ **using** *eudorus-dense-rational by blast*

hence $n\text{-}nonneg: n \geq 0$ **by** *simp*

{

fix $z :: \text{int}$ **assume** $z\text{-}nonneg: z \geq 0$

have $z * m = \lfloor \text{of-int } (z * n) * (\text{of-int } m / \text{of-int } n) \rfloor :: \text{real}$ **using** $*(3)$ **by** *simp (auto simp only: of-int-mult[symmetric] floor-of-int)*

also have $\dots \leq \lfloor \text{of-int } (z * n) * y \rfloor$ **using** $*(2)$ **by** (*meson floor-mono mult-left-mono n-nonneg nless-le of-int-nonneg z-nonneg zero-le-mult-iff*)

also have $\dots \leq u(z * n)$ **using** $u\text{-}Sup\text{-}nonneg[OF y(2)]$ *mult-nonneg-nonneg[OF z-nonneg n-nonneg]* **by** *blast*

finally have $u(z * n) \geq z * m$.

}
hence $abs-real (u *_{e} (*) n) \geq of-int m$ **using** *slope unfolding real-of-int eudoxus-times-def* **by** *(intro abs-real-leI[where ?N=0]) (auto simp add: mult.commute)*

moreover have $abs-real u * of-int n = abs-real (u *_{e} (*) n)$ **unfolding** *real-of-int* **using** *slope* **by** *(simp add: eudoxus-times-def comp-def)*

ultimately have $s \leq abs-real u$ **using** *** **by** *(metis leI mult-imp-div-pos-le of-int-0-less-iff order-le-less-trans order-less-asm)*

}
moreover
{
fix y **assume** *asm: $s \leq y$ if $s \in S$ for s*
assume $abs-real u > y$
then obtain $m n :: int$ **where** $*$: $y < (of-int m / of-int n) (of-int m / of-int n) < abs-real u n > 0$ **using** *eudoxus-dense-rational* **by** *blast*
hence $of-int m < abs-real u * of-int n$ **by** *(simp add: pos-divide-less-eq)*
hence $of-int m < abs-real (u *_{e} (*) n)$ **unfolding** *real-of-int* **using** *slope* **by** *(simp add: eudoxus-times-def comp-def)*
moreover have $slope (u *_{e} (*) n)$ **using** *slope* **by** *(simp add: eudoxus-times-def)*
ultimately obtain z **where** $z: (u *_{e} (*) n) z > m * z z \geq 1$ **unfolding** *real-of-int* **using** *abs-real-lessD* **by** *blast*
hence $** : u (n * z) > m * z$ **by** *(simp add: eudoxus-times-def comp-def)*

obtain x **where** $x: x \in S u (n * z) = \lfloor of-int (n * z) * x \rfloor$ **using** *u-mem[of n * z] zsgn-def[of n * z] mult-pos-pos[OF *(3), of z] z(2)* **by** *fastforce*

have $of-int (n * z) * x \leq of-int z * of-int n * y$ **using** *asm[OF x(1)]* **using** z **by** *auto*
also have $\dots < of-int z * of-int m$ **using** $*$ **by** *(simp add: mult.commute pos-less-divide-eq)*
finally have $of-int (n * z) * x < of-int (m * z)$ **by** *(simp add: mult.commute)*
hence *False* **using** $**$ **by** *(metis floor-less-iff less-le-not-le x(2))*

}
ultimately show *?thesis* **using** *that* **by** *force*
qed *blast*

end

References

- [1] R. D. Arthan. The Eudoxus real numbers. arXiv:math/0405454, 2004.