

# Isabelle/HOL-NSA — Non-Standard Analysis

May 22, 2012

## Contents

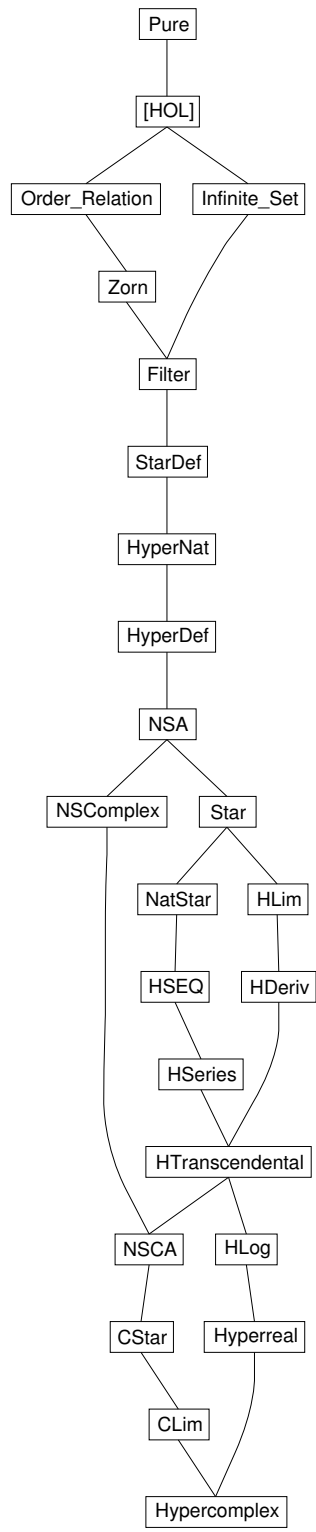
<b>1</b>	<b>Order-Relation: Orders as Relations</b>	<b>7</b>
1.1	Orders on a set . . . . .	7
1.2	Orders on the field . . . . .	8
1.3	Orders on a type . . . . .	8
<b>2</b>	<b>Zorn: Zorn's Lemma</b>	<b>8</b>
2.1	Mathematical Preamble . . . . .	9
2.2	Hausdorff's Theorem: Every Set Contains a Maximal Chain.	10
2.3	Zorn's Lemma: If All Chains Have Upper Bounds Then There Is a Maximal Element . . . . .	11
2.4	Alternative version of Zorn's Lemma . . . . .	11
<b>3</b>	<b>Infinite-Set: Infinite Sets and Related Concepts</b>	<b>13</b>
3.1	Infinite Sets . . . . .	13
3.2	Infinitely Many and Almost All . . . . .	16
3.3	Enumeration of an Infinite Set . . . . .	19
3.4	Miscellaneous . . . . .	19
<b>4</b>	<b>Filter: Filters and Ultrafilters</b>	<b>20</b>
4.1	Definitions and basic properties . . . . .	20
4.1.1	Filters . . . . .	20
4.1.2	Ultrafilters . . . . .	20
4.1.3	Free Ultrafilters . . . . .	21
4.2	Collect properties . . . . .	21
4.3	Maximal filter = Ultrafilter . . . . .	22
4.4	Ultrafilter Theorem . . . . .	22
4.4.1	Unions of chains of superfrechets . . . . .	23
4.4.2	Existence of free ultrafilter . . . . .	24

<b>5</b>	<b>StarDef: Construction of Star Types Using Ultrafilters</b>	<b>24</b>
5.1	A Free Ultrafilter over the Naturals . . . . .	25
5.2	Definition of <i>star</i> type constructor . . . . .	25
5.3	Transfer principle . . . . .	26
5.4	Standard elements . . . . .	27
5.5	Internal functions . . . . .	28
5.6	Internal predicates . . . . .	29
5.7	Internal sets . . . . .	30
5.8	Syntactic classes . . . . .	32
5.9	Ordering and lattice classes . . . . .	37
5.10	Ordered group classes . . . . .	38
5.11	Ring and field classes . . . . .	39
5.12	Power . . . . .	41
5.13	Number classes . . . . .	41
5.14	Finite class . . . . .	42
<b>6</b>	<b>HyperNat: Hypernatural numbers</b>	<b>43</b>
6.1	Properties Transferred from Naturals . . . . .	43
6.2	Properties of the set of embedded natural numbers . . . . .	45
6.3	Infinite Hypernatural Numbers – <i>HNatInfinite</i> . . . . .	46
6.3.1	Closure Rules . . . . .	46
6.4	Existence of an infinite hypernatural number . . . . .	47
6.4.1	Alternative characterization of the set of infinite hypernaturals . . . . .	48
6.4.2	Alternative Characterization of <i>HNatInfinite</i> using Free Ultrafilter . . . . .	48
6.5	Embedding of the Hypernaturals into other types . . . . .	49
<b>7</b>	<b>HyperDef: Construction of Hyperreals Using Ultrafilters</b>	<b>50</b>
7.1	Real vector class instances . . . . .	51
7.2	Injection from <i>hypreal</i> . . . . .	52
7.3	Properties of <i>starrel</i> . . . . .	53
7.4	<i>hypreal-of-real</i> : the Injection from <i>real</i> to <i>hypreal</i> . . . . .	53
7.5	Properties of <i>star-n</i> . . . . .	53
7.6	Misc Others . . . . .	54
7.7	Existence of Infinite Hyperreal Number . . . . .	54
7.8	Absolute Value Function for the Hyperreals . . . . .	55
7.9	Embedding the Naturals into the Hyperreals . . . . .	55
7.10	Exponentials on the Hyperreals . . . . .	56
7.11	Powers with Hypernatural Exponents . . . . .	57

<b>8 NSA: Infinite Numbers, Infinitesimals, Infinitely Close Relation</b>	<b>60</b>
8.1 Nonstandard Extension of the Norm Function . . . . .	61
8.2 Closure Laws for the Standard Reals . . . . .	64
8.3 Set of Finite Elements is a Subring of the Extended Reals . .	65
8.4 Set of Infinitesimals is a Subring of the Hyperreals . . . . .	66
8.5 The Infinitely Close Relation . . . . .	70
8.6 Zero is the Only Infinitesimal that is also a Real . . . . .	74
8.7 Uniqueness: Two Infinitely Close Reals are Equal . . . . .	76
8.8 Existence of Unique Real Infinitely Close . . . . .	77
8.8.1 Lifting of the Ub and Lub Properties . . . . .	77
8.9 Finite, Infinite and Infinitesimal . . . . .	80
8.10 Theorems about Monads . . . . .	83
8.11 Proof that $x \approx y$ implies $ x  \approx  y $ . . . . .	83
8.12 More <i>HFinite</i> and <i>Infinitesimal</i> Theorems . . . . .	85
8.13 Theorems about Standard Part . . . . .	86
8.14 Alternative Definitions using Free Ultrafilter . . . . .	89
8.14.1 <i>HFinite</i> . . . . .	89
8.14.2 <i>HInfinite</i> . . . . .	89
8.14.3 <i>Infinitesimal</i> . . . . .	90
8.15 Proof that $\omega$ is an infinite number . . . . .	90
<b>9 NSComplex: Nonstandard Complex Numbers</b>	<b>93</b>
9.1 Properties of Nonstandard Real and Imaginary Parts . . . . .	95
9.2 Addition for Nonstandard Complex Numbers . . . . .	96
9.3 More Minus Laws . . . . .	96
9.4 More Multiplication Laws . . . . .	96
9.5 Subtraction and Division . . . . .	97
9.6 Embedding Properties for <i>hcomplex-of-hypreal</i> Map . . . . .	97
9.7 HComplex theorems . . . . .	97
9.8 Modulus (Absolute Value) of Nonstandard Complex Number	97
9.9 Conjugation . . . . .	99
9.10 More Theorems about the Function <i>hcmmod</i> . . . . .	100
9.11 Exponentiation . . . . .	100
9.12 The Function <i>hsgn</i> . . . . .	101
9.13 Polar Form for Nonstandard Complex Numbers . . . . .	102
9.14 <i>hcomplex-of-complex</i> : the Injection from type <i>complex</i> to to <i>hcomplex</i> . . . . .	105
9.15 Numerals and Arithmetic . . . . .	105
<b>10 Star: Star-Transforms in Non-Standard Analysis</b>	<b>106</b>
10.1 Properties of the Star-transform Applied to Sets of Reals . .	107

<b>11 NatStar: Star-transforms for the Hypernaturals</b>	<b>112</b>
11.1 Nonstandard Extensions of Functions . . . . .	113
11.2 Nonstandard Characterization of Induction . . . . .	115
<b>12 HSEQ: Sequences and Convergence (Nonstandard)</b>	<b>116</b>
12.1 Limits of Sequences . . . . .	116
12.1.1 Equivalence of <i>LIMSEQ</i> and <i>NSLIMSEQ</i> . . . . .	119
12.1.2 Derived theorems about <i>NSLIMSEQ</i> . . . . .	119
12.2 Convergence . . . . .	120
12.3 Bounded Monotonic Sequences . . . . .	120
12.3.1 Upper Bounds and Lubs of Bounded Sequences . . . . .	121
12.3.2 A Bounded and Monotonic Sequence Converges . . . . .	121
12.4 Cauchy Sequences . . . . .	121
12.4.1 Equivalence Between NS and Standard . . . . .	122
12.4.2 Cauchy Sequences are Bounded . . . . .	122
12.4.3 Cauchy Sequences are Convergent . . . . .	122
12.5 Power Sequences . . . . .	123
<b>13 HSeries: Finite Summation and Infinite Series for Hyperreals</b>	<b>123</b>
13.1 Nonstandard Sums . . . . .	125
<b>14 HLim: Limits and Continuity (Nonstandard)</b>	<b>126</b>
14.1 Limits of Functions . . . . .	127
14.1.1 Equivalence of <i>LIM</i> and <i>NSLIM</i> . . . . .	129
14.2 Continuity . . . . .	129
14.3 Uniform Continuity . . . . .	130
<b>15 HDeriv: Differentiation (Nonstandard)</b>	<b>131</b>
15.1 Derivatives . . . . .	131
15.1.1 Equivalence of NS and Standard definitions . . . . .	135
15.1.2 Differentiability predicate . . . . .	136
15.2 (NS) Increment . . . . .	136
<b>16 HTranscendental: Nonstandard Extensions of Transcendental Functions</b>	<b>137</b>
16.1 Nonstandard Extension of Square Root Function . . . . .	137
<b>17 NSCA: Non-Standard Complex Analysis</b>	<b>144</b>
17.1 Closure Laws for SComplex, the Standard Complex Numbers	145
17.2 The Finite Elements form a Subring . . . . .	146
17.3 The Complex Infinitesimals form a Subring . . . . .	146
17.4 The “Infinitely Close” Relation . . . . .	147
17.5 Zero is the Only Infinitesimal Complex Number . . . . .	147
17.6 Properties of <i>hRe</i> , <i>hIm</i> and <i>HComplex</i> . . . . .	148

17.7 Theorems About Monads . . . . .	150
17.8 Theorems About Standard Part . . . . .	150
<b>18 CStar: Star-transforms in NSA, Extending Sets of Complex Numbers and Complex Functions</b>	<b>153</b>
18.1 Properties of the *-Transform Applied to Sets of Reals . . . .	153
18.2 Theorems about Nonstandard Extensions of Functions . . . .	153
18.3 Internal Functions - Some Redundancy With *f* Now . . . .	153
<b>19 CLim: Limits, Continuity and Differentiation for Complex Functions</b>	<b>154</b>
19.1 Limit of Complex to Complex Function . . . . .	154
19.2 Continuity . . . . .	155
19.3 Functions from Complex to Reals . . . . .	155
19.4 Differentiation of Natural Number Powers . . . . .	156
19.5 Derivative of Reciprocals (Function <i>inverse</i> ) . . . . .	156
19.6 Derivative of Quotient . . . . .	156
19.7 Caratheodory Formulation of Derivative at a Point: Standard Proof . . . . .	157
<b>20 HLog: Logarithms: Non-Standard Version</b>	<b>157</b>



# 1 Order-Relation: Orders as Relations

```
theory Order-Relation
imports Main
begin
```

## 1.1 Orders on a set

**definition** *preorder-on*  $A\ r \equiv \text{refl-on } A\ r \wedge \text{trans } r$

**definition** *partial-order-on*  $A\ r \equiv \text{preorder-on } A\ r \wedge \text{antisym } r$

**definition** *linear-order-on*  $A\ r \equiv \text{partial-order-on } A\ r \wedge \text{total-on } A\ r$

**definition** *strict-linear-order-on*  $A\ r \equiv \text{trans } r \wedge \text{irrefl } r \wedge \text{total-on } A\ r$

**definition** *well-order-on*  $A\ r \equiv \text{linear-order-on } A\ r \wedge \text{wf}(r - \text{Id})$

**lemmas** *order-on-defs* =  
*preorder-on-def partial-order-on-def linear-order-on-def*  
*strict-linear-order-on-def well-order-on-def*

**lemma** *preorder-on-empty[simp]*: *preorder-on*  $\{\}\ \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *partial-order-on-empty[simp]*: *partial-order-on*  $\{\}\ \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *linear-order-on-empty[simp]*: *linear-order-on*  $\{\}\ \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *well-order-on-empty[simp]*: *well-order-on*  $\{\}\ \{\}$   
 $\langle \text{proof} \rangle$

**lemma** *preorder-on-converse[simp]*: *preorder-on*  $A\ (r^{-1}) = \text{preorder-on } A\ r$   
 $\langle \text{proof} \rangle$

**lemma** *partial-order-on-converse[simp]*:  
*partial-order-on*  $A\ (r^{-1}) = \text{partial-order-on } A\ r$   
 $\langle \text{proof} \rangle$

**lemma** *linear-order-on-converse[simp]*:  
*linear-order-on*  $A\ (r^{-1}) = \text{linear-order-on } A\ r$   
 $\langle \text{proof} \rangle$

**lemma** *strict-linear-order-on-diff-Id*:  
*linear-order-on*  $A\ r \implies \text{strict-linear-order-on } A\ (r - \text{Id})$

*<proof>*

## 1.2 Orders on the field

**abbreviation** *Refl*  $r \equiv \text{refl-on (Field } r) r$

**abbreviation** *Preorder*  $r \equiv \text{preorder-on (Field } r) r$

**abbreviation** *Partial-order*  $r \equiv \text{partial-order-on (Field } r) r$

**abbreviation** *Total*  $r \equiv \text{total-on (Field } r) r$

**abbreviation** *Linear-order*  $r \equiv \text{linear-order-on (Field } r) r$

**abbreviation** *Well-order*  $r \equiv \text{well-order-on (Field } r) r$

**lemma** *subset-Image-Image-iff*:

$\llbracket \text{Preorder } r; A \subseteq \text{Field } r; B \subseteq \text{Field } r \rrbracket \implies$   
 $r \text{ “ } A \subseteq r \text{ “ } B \iff (\forall a \in A. \exists b \in B. (b,a):r)$

*<proof>*

**lemma** *subset-Image1-Image1-iff*:

$\llbracket \text{Preorder } r; a : \text{Field } r; b : \text{Field } r \rrbracket \implies r \text{ “ } \{a\} \subseteq r \text{ “ } \{b\} \iff (b,a):r$

*<proof>*

**lemma** *Refl-antisym-eq-Image1-Image1-iff*:

$\llbracket \text{Refl } r; \text{antisym } r; a:\text{Field } r; b:\text{Field } r \rrbracket \implies r \text{ “ } \{a\} = r \text{ “ } \{b\} \iff a=b$

*<proof>*

**lemma** *Partial-order-eq-Image1-Image1-iff*:

$\llbracket \text{Partial-order } r; a:\text{Field } r; b:\text{Field } r \rrbracket \implies r \text{ “ } \{a\} = r \text{ “ } \{b\} \iff a=b$

*<proof>*

## 1.3 Orders on a type

**abbreviation** *strict-linear-order*  $\equiv \text{strict-linear-order-on UNIV}$

**abbreviation** *linear-order*  $\equiv \text{linear-order-on UNIV}$

**abbreviation** *well-order*  $r \equiv \text{well-order-on UNIV}$

**end**

## 2 Zorn: Zorn’s Lemma

**theory** *Zorn*

**imports** *Order-Relation Main*

begin

**definition** *chain-subset* :: 'a set set  $\Rightarrow$  bool (*chain* $\subseteq$ )

**where**

$$\text{chain}\subseteq C \equiv \forall A \in C. \forall B \in C. A \subseteq B \vee B \subseteq A$$

The lemma and section numbers refer to an unpublished article [?].

**definition** *chain* :: 'a set set  $\Rightarrow$  'a set set set

**where**

$$\text{chain } S = \{F. F \subseteq S \wedge \text{chain}\subseteq F\}$$

**definition** *super* :: 'a set set  $\Rightarrow$  'a set set  $\Rightarrow$  'a set set set

**where**

$$\text{super } S c = \{d. d \in \text{chain } S \wedge c \subset d\}$$

**definition** *maxchain* :: 'a set set  $\Rightarrow$  'a set set set

**where**

$$\text{maxchain } S = \{c. c \in \text{chain } S \wedge \text{super } S c = \{\}\}$$

**definition** *succ* :: 'a set set  $\Rightarrow$  'a set set  $\Rightarrow$  'a set set

**where**

*succ*  $S c = (\text{if } c \notin \text{chain } S \vee c \in \text{maxchain } S \text{ then } c \text{ else SOME } c'. c' \in \text{super } S c)$

**inductive-set** *TFin* :: 'a set set  $\Rightarrow$  'a set set set

**for**  $S :: 'a \text{ set set}$

**where**

$$\text{succ}I: \quad x \in \text{TFin } S \implies \text{succ } S x \in \text{TFin } S$$

$$| \text{Pow-Union}I: Y \in \text{Pow } (\text{TFin } S) \implies \bigcup Y \in \text{TFin } S$$

## 2.1 Mathematical Preamble

**lemma** *Union-lemma0*:

$$(\forall x \in C. x \subseteq A \mid B \subseteq x) \implies \text{Union}(C) \subseteq A \mid B \subseteq \text{Union}(C)$$

*<proof>*

This is theorem *increasingD2* of ZF/Zorn.thy

**lemma** *Abrial-axiom1*:  $x \subseteq \text{succ } S x$

*<proof>*

**lemmas** *TFin-UnionI* = *TFin.Pow-UnionI* [*OF PowI*]

**lemma** *TFin-induct*:

**assumes**  $H: n \in \text{TFin } S$  **and**

$$I: !!x. x \in \text{TFin } S \implies P x \implies P (\text{succ } S x)$$

$$!!Y. Y \subseteq \text{TFin } S \implies \text{Ball } Y P \implies P (\text{Union } Y)$$

**shows**  $P n$

*<proof>*

**lemma** *succ-trans*:  $x \subseteq y \implies x \subseteq \text{succ } S y$

*<proof>*

Lemma 1 of section 3.1

**lemma** *TFin-linear-lemma1*:

$[[ n \in TFin S; m \in TFin S;$   
 $\quad \forall x \in TFin S. x \subseteq m \longrightarrow x = m \mid \text{succ } S x \subseteq m$

$]] \implies n \subseteq m \mid \text{succ } S m \subseteq n$

*<proof>*

Lemma 2 of section 3.2

**lemma** *TFin-linear-lemma2*:

$m \in TFin S \implies \forall n \in TFin S. n \subseteq m \longrightarrow n = m \mid \text{succ } S n \subseteq m$

*<proof>*

Re-ordering the premises of Lemma 2

**lemma** *TFin-subsetD*:

$[[ n \subseteq m; m \in TFin S; n \in TFin S ]] \implies n = m \mid \text{succ } S n \subseteq m$

*<proof>*

Consequences from section 3.3 – Property 3.2, the ordering is total

**lemma** *TFin-subset-linear*:  $[[ m \in TFin S; n \in TFin S ]] \implies n \subseteq m \mid m \subseteq n$

*<proof>*

Lemma 3 of section 3.3

**lemma** *eq-succ-upper*:  $[[ n \in TFin S; m \in TFin S; m = \text{succ } S m ]] \implies n \subseteq m$

*<proof>*

Property 3.3 of section 3.3

**lemma** *equal-succ-Union*:  $m \in TFin S \implies (m = \text{succ } S m) = (m = \text{Union}(TFin S))$

*<proof>*

## 2.2 Hausdorff’s Theorem: Every Set Contains a Maximal Chain.

NB: We assume the partial ordering is  $\subseteq$ , the subset relation!

**lemma** *empty-set-mem-chain*:  $(\{\} :: 'a \text{ set set}) \in \text{chain } S$

*<proof>*

**lemma** *super-subset-chain*:  $\text{super } S c \subseteq \text{chain } S$

*<proof>*

**lemma** *maxchain-subset-chain*:  $\text{maxchain } S \subseteq \text{chain } S$

*<proof>*

**lemma** *mem-super-Ex*:  $c \in \text{chain } S - \text{maxchain } S \implies \exists d. d \in \text{super } S c$   
 ⟨proof⟩

**lemma** *select-super*:

$c \in \text{chain } S - \text{maxchain } S \implies (\exists c'. c': \text{super } S c): \text{super } S c$   
 ⟨proof⟩

**lemma** *select-not-equals*:

$c \in \text{chain } S - \text{maxchain } S \implies (\exists c'. c': \text{super } S c) \neq c$   
 ⟨proof⟩

**lemma** *succI3*:  $c \in \text{chain } S - \text{maxchain } S \implies \text{succ } S c = (\exists c'. c': \text{super } S c)$   
 ⟨proof⟩

**lemma** *succ-not-equals*:  $c \in \text{chain } S - \text{maxchain } S \implies \text{succ } S c \neq c$   
 ⟨proof⟩

**lemma** *TFin-chain-lemma4*:  $c \in \text{TFin } S \implies (c :: 'a \text{ set set}): \text{chain } S$   
 ⟨proof⟩

**theorem** *Hausdorff*:  $\exists c. (c :: 'a \text{ set set}): \text{maxchain } S$   
 ⟨proof⟩

### 2.3 Zorn’s Lemma: If All Chains Have Upper Bounds Then There Is a Maximal Element

**lemma** *chain-extend*:

$[\![ c \in \text{chain } S; z \in S; \forall x \in c. x \subseteq (z :: 'a \text{ set}) ]\!] \implies \{z\} \cup c \in \text{chain } S$   
 ⟨proof⟩

**lemma** *chain-Union-upper*:  $[\![ c \in \text{chain } S; x \in c ]\!] \implies x \subseteq \text{Union}(c)$   
 ⟨proof⟩

**lemma** *chain-ball-Union-upper*:  $c \in \text{chain } S \implies \forall x \in c. x \subseteq \text{Union}(c)$   
 ⟨proof⟩

**lemma** *maxchain-Zorn*:

$[\![ c \in \text{maxchain } S; u \in S; \text{Union}(c) \subseteq u ]\!] \implies \text{Union}(c) = u$   
 ⟨proof⟩

**theorem** *Zorn-Lemma*:

$\forall c \in \text{chain } S. \text{Union}(c) \in S \implies \exists y \in S. \forall z \in S. y \subseteq z \longrightarrow y = z$   
 ⟨proof⟩

### 2.4 Alternative version of Zorn’s Lemma

**lemma** *Zorn-Lemma2*:

$\forall c \in \text{chain } S. \exists y \in S. \forall x \in c. x \subseteq y$

$\implies \exists y \in S. \forall x \in S. (y :: 'a \text{ set}) \subseteq x \implies y = x$   
 ⟨proof⟩

Various other lemmas

**lemma** *chainD*:  $[[ c \in \text{chain } S; x \in c; y \in c ]] \implies x \subseteq y \mid y \subseteq x$   
 ⟨proof⟩

**lemma** *chainD2*:  $!!(c :: 'a \text{ set set}). c \in \text{chain } S \implies c \subseteq S$   
 ⟨proof⟩

**definition** *Chain* ::  $('a * 'a) \text{ set} \Rightarrow 'a \text{ set set}$  **where**  
*Chain*  $r \equiv \{A. \forall a \in A. \forall b \in A. (a, b) : r \vee (b, a) \in r\}$

**lemma** *mono-Chain*:  $r \subseteq s \implies \text{Chain } r \subseteq \text{Chain } s$   
 ⟨proof⟩

Zorn’s lemma for partial orders:

**lemma** *Zorns-po-lemma*:

**assumes** *po*: *Partial-order*  $r$  **and**  $u$ :  $\forall C \in \text{Chain } r. \exists u \in \text{Field } r. \forall a \in C. (a, u) : r$

**shows**  $\exists m \in \text{Field } r. \forall a \in \text{Field } r. (m, a) : r \implies a = m$

⟨proof⟩

**definition** *init-seg-of* ::  $(( 'a * 'a) \text{ set} * ('a * 'a) \text{ set}) \text{ set}$  **where**  
*init-seg-of*  $\equiv \{(r, s). r \subseteq s \wedge (\forall a \ b \ c. (a, b) : s \wedge (b, c) : r \implies (a, b) : r)\}$

**abbreviation** *initialSegmentOf* ::  $('a * 'a) \text{ set} \Rightarrow ('a * 'a) \text{ set} \Rightarrow \text{bool}$

(**infix** *initial'-segment'-of* 55) **where**

$r$  *initial-segment-of*  $s \equiv (r, s) : \text{init-seg-of}$

**lemma** *refl-on-init-seg-of*[*simp*]:  $r$  *initial-segment-of*  $r$

⟨proof⟩

**lemma** *trans-init-seg-of*:

$r$  *initial-segment-of*  $s \implies s$  *initial-segment-of*  $t \implies r$  *initial-segment-of*  $t$

⟨proof⟩

**lemma** *antisym-init-seg-of*:

$r$  *initial-segment-of*  $s \implies s$  *initial-segment-of*  $r \implies r = s$

⟨proof⟩

**lemma** *Chain-init-seg-of-Union*:

$R \in \text{Chain}$  *init-seg-of*  $\implies r \in R \implies r$  *initial-segment-of*  $\bigcup R$

⟨proof⟩

**lemma** *chain-subset-trans-Union*:

$\text{chain}_{\subseteq} R \implies \forall r \in R. \text{trans } r \implies \text{trans}(\bigcup R)$

*<proof>*

**lemma** *chain-subset-antisym-Union:*

*chain*  $\subseteq$   $R \implies \forall r \in R. \text{antisym } r \implies \text{antisym}(\bigcup R)$   
*<proof>*

**lemma** *chain-subset-Total-Union:*

**assumes** *chain*  $\subseteq$   $R \ \forall r \in R. \text{Total } r$   
**shows** *Total*  $(\bigcup R)$   
*<proof>*

**lemma** *wf-Union-wf-init-segs:*

**assumes**  $R \in \text{Chain init-seg-of}$  **and**  $\forall r \in R. \text{wf } r$  **shows** *wf*  $(\bigcup R)$   
*<proof>*

**lemma** *initial-segment-of-Diff:*

*p initial-segment-of q*  $\implies p - s \text{ initial-segment-of } q - s$   
*<proof>*

**lemma** *Chain-inits-DiffI:*

$R \in \text{Chain init-seg-of} \implies \{r - s \mid r. r \in R\} \in \text{Chain init-seg-of}$   
*<proof>*

**theorem** *well-ordering:*  $\exists r::('a*'a)\text{set}. \text{Well-order } r \wedge \text{Field } r = \text{UNIV}$

*<proof>*

**corollary** *well-order-on:*  $\exists r::('a*'a)\text{set}. \text{well-order-on } A \ r$

*<proof>*

**end**

### 3 Infinite-Set: Infinite Sets and Related Concepts

**theory** *Infinite-Set*

**imports** *Main*

**begin**

#### 3.1 Infinite Sets

Some elementary facts about infinite sets, mostly by Stefan Merz. Beware! Because "infinite" merely abbreviates a negation, these lemmas may not work well with *blast*.

**abbreviation**

*infinite*  $:: 'a \text{ set} \Rightarrow \text{bool}$  **where**

*infinite*  $S == \neg \text{finite } S$

Infinite sets are non-empty, and if we remove some elements from an infinite

set, the result is still infinite.

**lemma** *infinite-imp-nonempty*:  $\text{infinite } S \implies S \neq \{\}$   
 ⟨proof⟩

**lemma** *infinite-remove*:  
 $\text{infinite } S \implies \text{infinite } (S - \{a\})$   
 ⟨proof⟩

**lemma** *Diff-infinite-finite*:  
**assumes**  $T$ : *finite*  $T$  **and**  $S$ : *infinite*  $S$   
**shows** *infinite*  $(S - T)$   
 ⟨proof⟩

**lemma** *Un-infinite*:  $\text{infinite } S \implies \text{infinite } (S \cup T)$   
 ⟨proof⟩

**lemma** *infinite-Un*:  $\text{infinite } (S \cup T) \longleftrightarrow \text{infinite } S \vee \text{infinite } T$   
 ⟨proof⟩

**lemma** *infinite-super*:  
**assumes**  $T$ :  $S \subseteq T$  **and**  $S$ : *infinite*  $S$   
**shows** *infinite*  $T$   
 ⟨proof⟩

As a concrete example, we prove that the set of natural numbers is infinite.

**lemma** *finite-nat-bounded*:  
**assumes**  $S$ : *finite*  $(S::\text{nat set})$   
**shows**  $\exists k. S \subseteq \{..<k\}$  (**is**  $\exists k. ?\text{bounded } S k$ )  
 ⟨proof⟩

**lemma** *finite-nat-iff-bounded*:  
 $\text{finite } (S::\text{nat set}) = (\exists k. S \subseteq \{..<k\})$  (**is**  $?lhs = ?rhs$ )  
 ⟨proof⟩

**lemma** *finite-nat-iff-bounded-le*:  
 $\text{finite } (S::\text{nat set}) = (\exists k. S \subseteq \{..k\})$  (**is**  $?lhs = ?rhs$ )  
 ⟨proof⟩

**lemma** *infinite-nat-iff-unbounded*:  
 $\text{infinite } (S::\text{nat set}) = (\forall m. \exists n. m < n \wedge n \in S)$   
 (**is**  $?lhs = ?rhs$ )  
 ⟨proof⟩

**lemma** *infinite-nat-iff-unbounded-le*:  
 $\text{infinite } (S::\text{nat set}) = (\forall m. \exists n. m \leq n \wedge n \in S)$   
 (**is**  $?lhs = ?rhs$ )  
 ⟨proof⟩

For a set of natural numbers to be infinite, it is enough to know that for any

number larger than some  $k$ , there is some larger number that is an element of the set.

**lemma** *unbounded-k-infinite*:

**assumes**  $k: \forall m. k < m \longrightarrow (\exists n. m < n \wedge n \in S)$

**shows** *infinite* ( $S::\text{nat set}$ )

*<proof>*

**lemma** *nat-infinite*: *infinite* ( $UNIV :: \text{nat set}$ )

*<proof>*

**lemma** *nat-not-finite*: *finite* ( $UNIV::\text{nat set}$ )  $\implies R$

*<proof>*

Every infinite set contains a countable subset. More precisely we show that a set  $S$  is infinite if and only if there exists an injective function from the naturals into  $S$ .

**lemma** *range-inj-infinite*:

*inj* ( $f::\text{nat} \Rightarrow 'a$ )  $\implies$  *infinite* (*range*  $f$ )

*<proof>*

**lemma** *int-infinite* [*simp*]:

**shows** *infinite* ( $UNIV::\text{int set}$ )

*<proof>*

The “only if” direction is harder because it requires the construction of a sequence of pairwise different elements of an infinite set  $S$ . The idea is to construct a sequence of non-empty and infinite subsets of  $S$  obtained by successively removing elements of  $S$ .

**lemma** *linorder-injI*:

**assumes** *hyp*:  $!!x y. x < (y::'a::\text{linorder}) \implies f x \neq f y$

**shows** *inj*  $f$

*<proof>*

**lemma** *infinite-countable-subset*:

**assumes** *inf*: *infinite* ( $S::'a \text{ set}$ )

**shows**  $\exists f. \text{inj } (f::\text{nat} \Rightarrow 'a) \wedge \text{range } f \subseteq S$

*<proof>*

**lemma** *infinite-iff-countable-subset*:

*infinite*  $S = (\exists f. \text{inj } (f::\text{nat} \Rightarrow 'a) \wedge \text{range } f \subseteq S)$

*<proof>*

For any function with infinite domain and finite range there is some element that is the image of infinitely many domain elements. In particular, any infinite sequence of elements from a finite set contains some element that occurs infinitely often.

**lemma** *inf-img-fin-dom*:  
**assumes** *img: finite (f'A) and dom: infinite A*  
**shows**  $\exists y \in f'A. \text{infinite } (f - \{y\})$   
*<proof>*

**lemma** *inf-img-fin-domE*:  
**assumes** *finite (f'A) and infinite A*  
**obtains** *y where y ∈ f'A and infinite (f - {y})*  
*<proof>*

### 3.2 Infinitely Many and Almost All

We often need to reason about the existence of infinitely many (resp., all but finitely many) objects satisfying some predicate, so we introduce corresponding binders and their proof rules.

**definition**  
*Inf-many* ::  $('a \Rightarrow \text{bool}) \Rightarrow \text{bool}$  (**binder** *INFM* 10) **where**  
*Inf-many*  $P = \text{infinite } \{x. P x\}$

**definition**  
*Alm-all* ::  $('a \Rightarrow \text{bool}) \Rightarrow \text{bool}$  (**binder** *MOST* 10) **where**  
*Alm-all*  $P = (\neg (\text{INFM } x. \neg P x))$

**notation** (*xsymbols*)  
*Inf-many* (**binder**  $\exists_{\infty}$  10) **and**  
*Alm-all* (**binder**  $\forall_{\infty}$  10)

**notation** (*HTML output*)  
*Inf-many* (**binder**  $\exists_{\infty}$  10) **and**  
*Alm-all* (**binder**  $\forall_{\infty}$  10)

**lemma** *INFM-iff-infinite*:  $(\text{INFM } x. P x) \longleftrightarrow \text{infinite } \{x. P x\}$   
*<proof>*

**lemma** *MOST-iff-cofinite*:  $(\text{MOST } x. P x) \longleftrightarrow \text{finite } \{x. \neg P x\}$   
*<proof>*

**lemmas** *MOST-iff-finiteNeg = MOST-iff-cofinite*

**lemma** *not-INFM [simp]*:  $\neg (\text{INFM } x. P x) \longleftrightarrow (\text{MOST } x. \neg P x)$   
*<proof>*

**lemma** *not-MOST [simp]*:  $\neg (\text{MOST } x. P x) \longleftrightarrow (\text{INFM } x. \neg P x)$   
*<proof>*

**lemma** *INFM-const [simp]*:  $(\text{INFM } x::'a. P) \longleftrightarrow P \wedge \text{infinite } (\text{UNIV}::'a \text{ set})$   
*<proof>*

**lemma** *MOST-const* [*simp*]:  $(MOST\ x::'a.\ P) \longleftrightarrow P \vee finite\ (UNIV::'a\ set)$   
 ⟨*proof*⟩

**lemma** *INFM-EX*:  $(\exists_{\infty}x.\ P\ x) \implies (\exists x.\ P\ x)$   
 ⟨*proof*⟩

**lemma** *ALL-MOST*:  $\forall x.\ P\ x \implies \forall_{\infty}x.\ P\ x$   
 ⟨*proof*⟩

**lemma** *INFM-E*: **assumes** *INFM*  $x.\ P\ x$  **obtains**  $x$  **where**  $P\ x$   
 ⟨*proof*⟩

**lemma** *MOST-I*: **assumes**  $\bigwedge x.\ P\ x$  **shows** *MOST*  $x.\ P\ x$   
 ⟨*proof*⟩

**lemma** *INFM-mono*:  
**assumes** *inf*:  $\exists_{\infty}x.\ P\ x$  **and**  $q: \bigwedge x.\ P\ x \implies Q\ x$   
**shows**  $\exists_{\infty}x.\ Q\ x$   
 ⟨*proof*⟩

**lemma** *MOST-mono*:  $\forall_{\infty}x.\ P\ x \implies (\bigwedge x.\ P\ x \implies Q\ x) \implies \forall_{\infty}x.\ Q\ x$   
 ⟨*proof*⟩

**lemma** *INFM-disj-distrib*:  
 $(\exists_{\infty}x.\ P\ x \vee Q\ x) \longleftrightarrow (\exists_{\infty}x.\ P\ x) \vee (\exists_{\infty}x.\ Q\ x)$   
 ⟨*proof*⟩

**lemma** *INFM-imp-distrib*:  
 $(INFM\ x.\ P\ x \longrightarrow Q\ x) \longleftrightarrow ((MOST\ x.\ P\ x) \longrightarrow (INFM\ x.\ Q\ x))$   
 ⟨*proof*⟩

**lemma** *MOST-conj-distrib*:  
 $(\forall_{\infty}x.\ P\ x \wedge Q\ x) \longleftrightarrow (\forall_{\infty}x.\ P\ x) \wedge (\forall_{\infty}x.\ Q\ x)$   
 ⟨*proof*⟩

**lemma** *MOST-conjI*:  
 $MOST\ x.\ P\ x \implies MOST\ x.\ Q\ x \implies MOST\ x.\ P\ x \wedge Q\ x$   
 ⟨*proof*⟩

**lemma** *INFM-conjI*:  
 $INFM\ x.\ P\ x \implies MOST\ x.\ Q\ x \implies INFM\ x.\ P\ x \wedge Q\ x$   
 ⟨*proof*⟩

**lemma** *MOST-rev-mp*:  
**assumes**  $\forall_{\infty}x.\ P\ x$  **and**  $\forall_{\infty}x.\ P\ x \longrightarrow Q\ x$   
**shows**  $\forall_{\infty}x.\ Q\ x$   
 ⟨*proof*⟩

**lemma** *MOST-imp-iff*:

**assumes**  $MOST\ x.\ P\ x$   
**shows**  $(MOST\ x.\ P\ x \longrightarrow Q\ x) \longleftrightarrow (MOST\ x.\ Q\ x)$   
 ⟨proof⟩

**lemma** *INFM-MOST-simps* [simp]:

$\bigwedge P\ Q.\ (INFM\ x.\ P\ x \wedge Q) \longleftrightarrow (INFM\ x.\ P\ x) \wedge Q$   
 $\bigwedge P\ Q.\ (INFM\ x.\ P \wedge Q\ x) \longleftrightarrow P \wedge (INFM\ x.\ Q\ x)$   
 $\bigwedge P\ Q.\ (MOST\ x.\ P\ x \vee Q) \longleftrightarrow (MOST\ x.\ P\ x) \vee Q$   
 $\bigwedge P\ Q.\ (MOST\ x.\ P \vee Q\ x) \longleftrightarrow P \vee (MOST\ x.\ Q\ x)$   
 $\bigwedge P\ Q.\ (MOST\ x.\ P\ x \longrightarrow Q) \longleftrightarrow ((INFM\ x.\ P\ x) \longrightarrow Q)$   
 $\bigwedge P\ Q.\ (MOST\ x.\ P \longrightarrow Q\ x) \longleftrightarrow (P \longrightarrow (MOST\ x.\ Q\ x))$   
 ⟨proof⟩

Properties of quantifiers with injective functions.

**lemma** *INFM-inj*:

$INFM\ x.\ P\ (f\ x) \implies inj\ f \implies INFM\ x.\ P\ x$   
 ⟨proof⟩

**lemma** *MOST-inj*:

$MOST\ x.\ P\ x \implies inj\ f \implies MOST\ x.\ P\ (f\ x)$   
 ⟨proof⟩

Properties of quantifiers with singletons.

**lemma** *not-INFM-eq* [simp]:

$\neg (INFM\ x.\ x = a)$   
 $\neg (INFM\ x.\ a = x)$   
 ⟨proof⟩

**lemma** *MOST-neq* [simp]:

$MOST\ x.\ x \neq a$   
 $MOST\ x.\ a \neq x$   
 ⟨proof⟩

**lemma** *INFM-neq* [simp]:

$(INFM\ x::'a.\ x \neq a) \longleftrightarrow infinite\ (UNIV::'a\ set)$   
 $(INFM\ x::'a.\ a \neq x) \longleftrightarrow infinite\ (UNIV::'a\ set)$   
 ⟨proof⟩

**lemma** *MOST-eq* [simp]:

$(MOST\ x::'a.\ x = a) \longleftrightarrow finite\ (UNIV::'a\ set)$   
 $(MOST\ x::'a.\ a = x) \longleftrightarrow finite\ (UNIV::'a\ set)$   
 ⟨proof⟩

**lemma** *MOST-eq-imp*:

$MOST\ x.\ x = a \longrightarrow P\ x$   
 $MOST\ x.\ a = x \longrightarrow P\ x$   
 ⟨proof⟩

Properties of quantifiers over the naturals.

**lemma** *INFM-nat*:  $(\exists_{\infty} n. P (n::nat)) = (\forall m. \exists n. m < n \wedge P n)$   
 ⟨proof⟩

**lemma** *INFM-nat-le*:  $(\exists_{\infty} n. P (n::nat)) = (\forall m. \exists n. m \leq n \wedge P n)$   
 ⟨proof⟩

**lemma** *MOST-nat*:  $(\forall_{\infty} n. P (n::nat)) = (\exists m. \forall n. m < n \longrightarrow P n)$   
 ⟨proof⟩

**lemma** *MOST-nat-le*:  $(\forall_{\infty} n. P (n::nat)) = (\exists m. \forall n. m \leq n \longrightarrow P n)$   
 ⟨proof⟩

### 3.3 Enumeration of an Infinite Set

The set’s element type must be wellordered (e.g. the natural numbers).

**primrec** (*in wellorder*) *enumerate* :: ‘a set  $\Rightarrow$  nat  $\Rightarrow$  ‘a **where**  
*enumerate-0*:  $enumerate\ S\ 0 = (LEAST\ n.\ n \in S)$   
 | *enumerate-Suc*:  $enumerate\ S\ (Suc\ n) = enumerate\ (S - \{LEAST\ n.\ n \in S\})\ n$

**lemma** *enumerate-Suc'*:  
 $enumerate\ S\ (Suc\ n) = enumerate\ (S - \{enumerate\ S\ 0\})\ n$   
 ⟨proof⟩

**lemma** *enumerate-in-set*:  $infinite\ S \Longrightarrow enumerate\ S\ n : S$   
 ⟨proof⟩

**declare** *enumerate-0* [*simp del*] *enumerate-Suc* [*simp del*]

**lemma** *enumerate-step*:  $infinite\ S \Longrightarrow enumerate\ S\ n < enumerate\ S\ (Suc\ n)$   
 ⟨proof⟩

**lemma** *enumerate-mono*:  $m < n \Longrightarrow infinite\ S \Longrightarrow enumerate\ S\ m < enumerate\ S\ n$   
 ⟨proof⟩

### 3.4 Miscellaneous

A few trivial lemmas about sets that contain at most one element. These simplify the reasoning about deterministic automata.

**definition**  
*atmost-one* :: ‘a set  $\Rightarrow$  bool **where**  
*atmost-one*  $S = (\forall x\ y. x \in S \wedge y \in S \longrightarrow x = y)$

**lemma** *atmost-one-empty*:  $S = \{\} \Longrightarrow atmost-one\ S$   
 ⟨proof⟩

**lemma** *atmost-one-singleton*:  $S = \{x\} \Longrightarrow atmost-one\ S$

*<proof>*

**lemma** *atmost-one-unique* [elim]: *atmost-one*  $S \implies x \in S \implies y \in S \implies y = x$   
*<proof>*

**end**

## 4 Filter: Filters and Ultrafilters

**theory** *Filter*

**imports** *~~/src/HOL/Library/Zorn* *~~/src/HOL/Library/Infinite-Set*

**begin**

### 4.1 Definitions and basic properties

#### 4.1.1 Filters

**locale** *filter* =

**fixes**  $F :: 'a \text{ set set}$

**assumes** *UNIV* [iff]:  $UNIV \in F$

**assumes** *empty* [iff]:  $\{\} \notin F$

**assumes** *Int*:  $\llbracket u \in F; v \in F \rrbracket \implies u \cap v \in F$

**assumes** *subset*:  $\llbracket u \in F; u \subseteq v \rrbracket \implies v \in F$

**begin**

**lemma** *memD*:  $A \in F \implies \neg A \notin F$   
*<proof>*

**lemma** *not-memI*:  $\neg A \in F \implies A \notin F$   
*<proof>*

**lemma** *Int-iff*:  $(x \cap y \in F) = (x \in F \wedge y \in F)$   
*<proof>*

**end**

#### 4.1.2 Ultrafilters

**locale** *ultrafilter* = *filter* +

**assumes** *ultra*:  $A \in F \vee \neg A \in F$

**begin**

**lemma** *memI*:  $\neg A \notin F \implies A \in F$   
*<proof>*

**lemma** *not-memD*:  $A \notin F \implies \neg A \in F$   
*<proof>*

**lemma** *not-mem-iff*:  $(A \notin F) = (\neg A \in F)$   
 ⟨proof⟩

**lemma** *Compl-iff*:  $(\neg A \in F) = (A \notin F)$   
 ⟨proof⟩

**lemma** *Un-iff*:  $(x \cup y \in F) = (x \in F \vee y \in F)$   
 ⟨proof⟩

**end**

### 4.1.3 Free Ultrafilters

**locale** *freeultrafilter* = *ultrafilter* +  
**assumes** *infinite*:  $A \in F \implies \text{infinite } A$   
**begin**

**lemma** *finite*:  $\text{finite } A \implies A \notin F$   
 ⟨proof⟩

**lemma** *singleton*:  $\{x\} \notin F$   
 ⟨proof⟩

**lemma** *insert-iff* [*simp*]:  $(\text{insert } x \ A \in F) = (A \in F)$   
 ⟨proof⟩

**lemma** *filter*: *filter*  $F$  ⟨proof⟩

**lemma** *ultrafilter*: *ultrafilter*  $F$  ⟨proof⟩

**end**

## 4.2 Collect properties

**lemma** (**in** *filter*) *Collect-ex*:  
 $(\{n. \exists x. P \ n \ x\} \in F) = (\exists X. \{n. P \ n \ (X \ n)\} \in F)$   
 ⟨proof⟩

**lemma** (**in** *filter*) *Collect-conj*:  
 $(\{n. P \ n \ \wedge \ Q \ n\} \in F) = (\{n. P \ n\} \in F \ \wedge \ \{n. Q \ n\} \in F)$   
 ⟨proof⟩

**lemma** (**in** *ultrafilter*) *Collect-not*:  
 $(\{n. \neg P \ n\} \in F) = (\{n. P \ n\} \notin F)$   
 ⟨proof⟩

**lemma** (**in** *ultrafilter*) *Collect-disj*:  
 $(\{n. P \ n \ \vee \ Q \ n\} \in F) = (\{n. P \ n\} \in F \ \vee \ \{n. Q \ n\} \in F)$   
 ⟨proof⟩

**lemma** (in *ultrafilter*) *Collect-all*:

$(\{n. \forall x. P n x\} \in F) = (\forall X. \{n. P n (X n)\} \in F)$   
 ⟨proof⟩

### 4.3 Maximal filter = Ultrafilter

A filter  $F$  is an ultrafilter iff it is a maximal filter, i.e. whenever  $G$  is a filter and  $F \subseteq G$  then  $F = G$

Lemmas that shows existence of an extension to what was assumed to be a maximal filter. Will be used to derive contradiction in proof of property of ultrafilter.

**lemma** *extend-lemma1*:  $UNIV \in F \implies A \in \{X. \exists f \in F. A \cap f \subseteq X\}$   
 ⟨proof⟩

**lemma** *extend-lemma2*:  $F \subseteq \{X. \exists f \in F. A \cap f \subseteq X\}$   
 ⟨proof⟩

**lemma** (in *filter*) *extend-filter*:

**assumes**  $A: - A \notin F$

**shows** *filter*  $\{X. \exists f \in F. A \cap f \subseteq X\}$  (is *filter* ? $X$ )

⟨proof⟩

**lemma** (in *filter*) *max-filter-ultrafilter*:

**assumes** *max*:  $\bigwedge G. \llbracket \text{filter } G; F \subseteq G \rrbracket \implies F = G$

**shows** *ultrafilter-axioms*  $F$

⟨proof⟩

**lemma** (in *ultrafilter*) *max-filter*:

**assumes**  $G$ : *filter*  $G$  and *sub*:  $F \subseteq G$  **shows**  $F = G$

⟨proof⟩

### 4.4 Ultrafilter Theorem

A local context makes proof of ultrafilter Theorem more modular

**context**

**fixes** *frechet* :: 'a set set

**and** *superfrechet* :: 'a set set set

**assumes** *infinite-UNIV*: *infinite* ( $UNIV :: 'a \text{ set}$ )

**defines** *frechet-def*:  $\text{frechet} \equiv \{A. \text{finite } (- A)\}$

**and** *superfrechet-def*:  $\text{superfrechet} \equiv \{G. \text{filter } G \wedge \text{frechet} \subseteq G\}$

**begin**

**lemma** *superfrechetI*:

$\llbracket \text{filter } G; \text{frechet} \subseteq G \rrbracket \implies G \in \text{superfrechet}$

*<proof>*

**lemma** *superfrechetD1*:

$G \in \text{superfrechet} \implies \text{filter } G$

*<proof>*

**lemma** *superfrechetD2*:

$G \in \text{superfrechet} \implies \text{frechet} \subseteq G$

*<proof>*

A few properties of free filters

**lemma** *filter-cofinite*:

**assumes** *inf*: *infinite* (*UNIV* :: 'a set)

**shows** *filter* {*A*:: 'a set. *finite* ( $- A$ )} (**is filter** ?*F*)

*<proof>*

We prove: 1. Existence of maximal filter i.e. ultrafilter; 2. Freeness property i.e ultrafilter is free. Use a locale to prove various lemmas and then export main result: The ultrafilter Theorem

**lemma** *filter-frechet*: *filter frechet*

*<proof>*

**lemma** *frechet-in-superfrechet*: *frechet*  $\in$  *superfrechet*

*<proof>*

**lemma** *lemma-mem-chain-filter*:

$\llbracket c \in \text{chain } \text{superfrechet}; x \in c \rrbracket \implies \text{filter } x$

*<proof>*

#### 4.4.1 Unions of chains of superfrechets

In this section we prove that superfrechet is closed with respect to unions of non-empty chains. We must show 1) Union of a chain is a filter, 2) Union of a chain contains frechet.

Number 2 is trivial, but 1 requires us to prove all the filter rules.

**lemma** *Union-chain-UNIV*:

$\llbracket c \in \text{chain } \text{superfrechet}; c \neq \{\} \rrbracket \implies \text{UNIV} \in \bigcup c$

*<proof>*

**lemma** *Union-chain-empty*:

$c \in \text{chain } \text{superfrechet} \implies \{\} \notin \bigcup c$

*<proof>*

**lemma** *Union-chain-Int*:

$\llbracket c \in \text{chain } \text{superfrechet}; u \in \bigcup c; v \in \bigcup c \rrbracket \implies u \cap v \in \bigcup c$

*<proof>*

**lemma** *Union-chain-subset*:

$\llbracket c \in \text{chain superfrechet}; u \in \bigcup c; u \subseteq v \rrbracket \implies v \in \bigcup c$   
 ⟨proof⟩

**lemma** *Union-chain-filter*:

**assumes** *chain*:  $c \in \text{chain superfrechet}$  **and** *nonempty*:  $c \neq \{\}$

**shows** *filter*  $(\bigcup c)$

⟨proof⟩

**lemma** *lemma-mem-chain-frechet-subset*:

$\llbracket c \in \text{chain superfrechet}; x \in c \rrbracket \implies \text{frechet} \subseteq x$   
 ⟨proof⟩

**lemma** *Union-chain-superfrechet*:

$\llbracket c \neq \{\}; c \in \text{chain superfrechet} \rrbracket \implies \bigcup c \in \text{superfrechet}$   
 ⟨proof⟩

#### 4.4.2 Existence of free ultrafilter

**lemma** *max-cofinite-filter-Ex*:

$\exists U \in \text{superfrechet}. \forall G \in \text{superfrechet}. U \subseteq G \longrightarrow U = G$   
 ⟨proof⟩

**lemma** *mem-superfrechet-all-infinite*:

$\llbracket U \in \text{superfrechet}; A \in U \rrbracket \implies \text{infinite } A$   
 ⟨proof⟩

There exists a free ultrafilter on any infinite set

**lemma** *freeultrafilter-Ex*:

$\exists U :: 'a \text{ set set}. \text{freeultrafilter } U$   
 ⟨proof⟩

**end**

**hide-const** (open) *filter*

**end**

## 5 StarDef: Construction of Star Types Using Ultrafilters

**theory** *StarDef*

**imports** *Filter*

**uses** (*transfer.ML*)

**begin**

## 5.1 A Free Ultrafilter over the Naturals

### definition

$FreeUltrafilterNat :: nat \ set \ set \ (\mathcal{U})$  **where**  
 $\mathcal{U} = (SOME\ U.\ freeultrafilter\ U)$

**lemma** *freeultrafilter-FreeUltrafilterNat*: *freeultrafilter*  $\mathcal{U}$   
 $\langle proof \rangle$

**interpretation** *FreeUltrafilterNat*: *freeultrafilter* *FreeUltrafilterNat*  
 $\langle proof \rangle$

This rule takes the place of the old ultra tactic

### lemma *ultra*:

$\llbracket \{n.\ P\ n\} \in \mathcal{U}; \{n.\ P\ n \longrightarrow Q\ n\} \in \mathcal{U} \rrbracket \implies \{n.\ Q\ n\} \in \mathcal{U}$   
 $\langle proof \rangle$

## 5.2 Definition of *star* type constructor

### definition

$starrel :: ((nat \Rightarrow 'a) \times (nat \Rightarrow 'a)) \ set$  **where**  
 $starrel = \{(X, Y).\ \{n.\ X\ n = Y\ n\} \in \mathcal{U}\}$

**definition**  $star = (UNIV :: (nat \Rightarrow 'a) \ set) // starrel$

**typedef** (open)  $'a\ star = star :: (nat \Rightarrow 'a) \ set \ set$   
 $\langle proof \rangle$

### definition

$star-n :: (nat \Rightarrow 'a) \Rightarrow 'a\ star$  **where**  
 $star-n\ X = Abs-star\ (starrel\ \{\{X\}\})$

**theorem** *star-cases* [*case-names* *star-n*, *cases type*: *star*]:  
 $(\bigwedge X.\ x = star-n\ X \implies P) \implies P$   
 $\langle proof \rangle$

**lemma** *all-star-eq*:  $(\forall x.\ P\ x) = (\forall X.\ P\ (star-n\ X))$   
 $\langle proof \rangle$

**lemma** *ex-star-eq*:  $(\exists x.\ P\ x) = (\exists X.\ P\ (star-n\ X))$   
 $\langle proof \rangle$

Proving that *starrel* is an equivalence relation

**lemma** *starrel-iff* [*iff*]:  $((X, Y) \in starrel) = (\{n.\ X\ n = Y\ n\} \in \mathcal{U})$   
 $\langle proof \rangle$

**lemma** *equiv-starrel*: *equiv* *UNIV* *starrel*  
 $\langle proof \rangle$

**lemmas** *equiv-starrel-iff* =

*eq-equiv-class-iff* [OF equiv-starrel UNIV-I UNIV-I]

**lemma** *starrel-in-star*:  $\text{starrel}^{\text{“}}\{x\} \in \text{star}$   
 ⟨proof⟩

**lemma** *star-n-eq-iff*:  $(\text{star-n } X = \text{star-n } Y) = (\{n. X n = Y n\} \in \mathcal{U})$   
 ⟨proof⟩

### 5.3 Transfer principle

This introduction rule starts each transfer proof.

**lemma** *transfer-start*:  
 $P \equiv \{n. Q\} \in \mathcal{U} \implies \text{Trueprop } P \equiv \text{Trueprop } Q$   
 ⟨proof⟩

Initialize transfer tactic.

⟨ML⟩

Transfer introduction rules.

**lemma** *transfer-ex* [transfer-intro]:  
 $\llbracket \bigwedge X. p (\text{star-n } X) \equiv \{n. P n (X n)\} \in \mathcal{U} \rrbracket$   
 $\implies \exists x::'a \text{ star. } p x \equiv \{n. \exists x. P n x\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-all* [transfer-intro]:  
 $\llbracket \bigwedge X. p (\text{star-n } X) \equiv \{n. P n (X n)\} \in \mathcal{U} \rrbracket$   
 $\implies \forall x::'a \text{ star. } p x \equiv \{n. \forall x. P n x\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-not* [transfer-intro]:  
 $\llbracket p \equiv \{n. P n\} \in \mathcal{U} \rrbracket \implies \neg p \equiv \{n. \neg P n\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-conj* [transfer-intro]:  
 $\llbracket p \equiv \{n. P n\} \in \mathcal{U}; q \equiv \{n. Q n\} \in \mathcal{U} \rrbracket$   
 $\implies p \wedge q \equiv \{n. P n \wedge Q n\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-disj* [transfer-intro]:  
 $\llbracket p \equiv \{n. P n\} \in \mathcal{U}; q \equiv \{n. Q n\} \in \mathcal{U} \rrbracket$   
 $\implies p \vee q \equiv \{n. P n \vee Q n\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-imp* [transfer-intro]:  
 $\llbracket p \equiv \{n. P n\} \in \mathcal{U}; q \equiv \{n. Q n\} \in \mathcal{U} \rrbracket$   
 $\implies p \longrightarrow q \equiv \{n. P n \longrightarrow Q n\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *transfer-iff* [transfer-intro]:

$$\begin{aligned} & \llbracket p \equiv \{n. P n\} \in \mathcal{U}; q \equiv \{n. Q n\} \in \mathcal{U} \rrbracket \\ & \implies p = q \equiv \{n. P n = Q n\} \in \mathcal{U} \\ \langle \text{proof} \rangle \end{aligned}$$

**lemma** *transfer-if-bool* [*transfer-intro*]:  

$$\begin{aligned} & \llbracket p \equiv \{n. P n\} \in \mathcal{U}; x \equiv \{n. X n\} \in \mathcal{U}; y \equiv \{n. Y n\} \in \mathcal{U} \rrbracket \\ & \implies (\text{if } p \text{ then } x \text{ else } y) \equiv \{n. \text{if } P n \text{ then } X n \text{ else } Y n\} \in \mathcal{U} \end{aligned}$$
 $\langle \text{proof} \rangle$

**lemma** *transfer-eq* [*transfer-intro*]:  

$$\llbracket x \equiv \text{star-}n X; y \equiv \text{star-}n Y \rrbracket \implies x = y \equiv \{n. X n = Y n\} \in \mathcal{U}$$
 $\langle \text{proof} \rangle$

**lemma** *transfer-if* [*transfer-intro*]:  

$$\begin{aligned} & \llbracket p \equiv \{n. P n\} \in \mathcal{U}; x \equiv \text{star-}n X; y \equiv \text{star-}n Y \rrbracket \\ & \implies (\text{if } p \text{ then } x \text{ else } y) \equiv \text{star-}n (\lambda n. \text{if } P n \text{ then } X n \text{ else } Y n) \end{aligned}$$
 $\langle \text{proof} \rangle$

**lemma** *transfer-fun-eq* [*transfer-intro*]:  

$$\begin{aligned} & \llbracket \bigwedge X. f (\text{star-}n X) = g (\text{star-}n X) \rrbracket \\ & \equiv \{n. F n (X n) = G n (X n)\} \in \mathcal{U} \\ & \implies f = g \equiv \{n. F n = G n\} \in \mathcal{U} \end{aligned}$$
 $\langle \text{proof} \rangle$

**lemma** *transfer-star-n* [*transfer-intro*]:  $\text{star-}n X \equiv \text{star-}n (\lambda n. X n)$   
 $\langle \text{proof} \rangle$

**lemma** *transfer-bool* [*transfer-intro*]:  $p \equiv \{n. p\} \in \mathcal{U}$   
 $\langle \text{proof} \rangle$

## 5.4 Standard elements

**definition**  
*star-of* :: 'a  $\Rightarrow$  'a **star where**  
*star-of* x == *star-n* ( $\lambda n. x$ )

**definition**  
*Standard* :: 'a **star set where**  
*Standard* = range *star-of*

Transfer tactic should remove occurrences of *star-of*  
 $\langle ML \rangle$

**declare** *star-of-def* [*transfer-intro*]

**lemma** *star-of-inject*:  $(\text{star-of } x = \text{star-of } y) = (x = y)$   
 $\langle \text{proof} \rangle$

**lemma** *Standard-star-of* [*simp*]:  $\text{star-of } x \in \text{Standard}$

*<proof>*

## 5.5 Internal functions

### definition

$\text{Ifun} :: ('a \Rightarrow 'b) \text{star} \Rightarrow 'a \text{star} \Rightarrow 'b \text{star} \text{ (-} \star \text{- [300,301] 300)}$  **where**  
 $\text{Ifun } f \equiv \lambda x. \text{Abs-star}$   
 $(\bigcup F \in \text{Rep-star } f. \bigcup X \in \text{Rep-star } x. \text{starrel}''\{\lambda n. F n (X n)\})$

### lemma *Ifun-congruent2*:

$\text{congruent2 starrel starrel } (\lambda F X. \text{starrel}''\{\lambda n. F n (X n)\})$   
*<proof>*

### lemma *Ifun-star-n*: $\text{star-n } F \star \text{star-n } X = \text{star-n } (\lambda n. F n (X n))$

*<proof>*

Transfer tactic should remove occurrences of *Ifun*

*<ML>*

### lemma *transfer-Ifun* [*transfer-intro*]:

$\llbracket f \equiv \text{star-n } F; x \equiv \text{star-n } X \rrbracket \Longrightarrow f \star x \equiv \text{star-n } (\lambda n. F n (X n))$   
*<proof>*

### lemma *Ifun-star-of* [*simp*]: $\text{star-of } f \star \text{star-of } x = \text{star-of } (f x)$

*<proof>*

### lemma *Standard-Ifun* [*simp*]:

$\llbracket f \in \text{Standard}; x \in \text{Standard} \rrbracket \Longrightarrow f \star x \in \text{Standard}$   
*<proof>*

Nonstandard extensions of functions

### definition

$\text{starfun} :: ('a \Rightarrow 'b) \Rightarrow ('a \text{star} \Rightarrow 'b \text{star}) \text{ (*f* - [80] 80)}$  **where**  
 $\text{starfun } f == \lambda x. \text{star-of } f \star x$

### definition

$\text{starfun2} :: ('a \Rightarrow 'b \Rightarrow 'c) \Rightarrow ('a \text{star} \Rightarrow 'b \text{star} \Rightarrow 'c \text{star})$   
 $(\text{*f2* - [80] 80})$  **where**  
 $\text{starfun2 } f == \lambda x y. \text{star-of } f \star x \star y$

**declare** *starfun-def* [*transfer-unfold*]

**declare** *starfun2-def* [*transfer-unfold*]

### lemma *starfun-star-n*: $(\text{*f* } f) (\text{star-n } X) = \text{star-n } (\lambda n. f (X n))$

*<proof>*

### lemma *starfun2-star-n*:

$(\text{*f2* } f) (\text{star-n } X) (\text{star-n } Y) = \text{star-n } (\lambda n. f (X n) (Y n))$   
*<proof>*

**lemma** *starfun-star-of* [simp]:  $(** f) (\text{star-of } x) = \text{star-of } (f x)$   
 ⟨proof⟩

**lemma** *starfun2-star-of* [simp]:  $(*f2* f) (\text{star-of } x) = ** f x$   
 ⟨proof⟩

**lemma** *Standard-starfun* [simp]:  $x \in \text{Standard} \implies \text{starfun } f x \in \text{Standard}$   
 ⟨proof⟩

**lemma** *Standard-starfun2* [simp]:  
 $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies \text{starfun2 } f x y \in \text{Standard}$   
 ⟨proof⟩

**lemma** *Standard-starfun-iff*:  
 assumes *inj*:  $\bigwedge x y. f x = f y \implies x = y$   
 shows  $(\text{starfun } f x \in \text{Standard}) = (x \in \text{Standard})$   
 ⟨proof⟩

**lemma** *Standard-starfun2-iff*:  
 assumes *inj*:  $\bigwedge a b a' b'. f a b = f a' b' \implies a = a' \wedge b = b'$   
 shows  $(\text{starfun2 } f x y \in \text{Standard}) = (x \in \text{Standard} \wedge y \in \text{Standard})$   
 ⟨proof⟩

## 5.6 Internal predicates

**definition** *unstar* ::  $\text{bool } \text{star} \Rightarrow \text{bool}$  **where**  
 $\text{unstar } b \longleftrightarrow b = \text{star-of } \text{True}$

**lemma** *unstar-star-n*:  $\text{unstar } (\text{star-n } P) = (\{n. P n\} \in \mathcal{U})$   
 ⟨proof⟩

**lemma** *unstar-star-of* [simp]:  $\text{unstar } (\text{star-of } p) = p$   
 ⟨proof⟩

Transfer tactic should remove occurrences of *unstar*  
 ⟨ML⟩

**lemma** *transfer-unstar* [transfer-intro]:  
 $p \equiv \text{star-n } P \implies \text{unstar } p \equiv \{n. P n\} \in \mathcal{U}$   
 ⟨proof⟩

**definition**  
 $\text{starP} :: ('a \Rightarrow \text{bool}) \Rightarrow 'a \text{ star} \Rightarrow \text{bool}$  (*\*p\** - [80] 80) **where**  
 $*p* P = (\lambda x. \text{unstar } (\text{star-of } P \star x))$

**definition**  
 $\text{starP2} :: ('a \Rightarrow 'b \Rightarrow \text{bool}) \Rightarrow 'a \text{ star} \Rightarrow 'b \text{ star} \Rightarrow \text{bool}$  (*\*p2\** - [80] 80) **where**  
 $*p2* P = (\lambda x y. \text{unstar } (\text{star-of } P \star x \star y))$

**declare** *starP-def* [*transfer-unfold*]

**declare** *starP2-def* [*transfer-unfold*]

**lemma** *starP-star-n*:  $( *p* P ) ( \text{star-n } X ) = ( \{n. P (X n)\} \in \mathcal{U} )$   
 $\langle \text{proof} \rangle$

**lemma** *starP2-star-n*:

$( *p2* P ) ( \text{star-n } X ) ( \text{star-n } Y ) = ( \{n. P (X n) (Y n)\} \in \mathcal{U} )$   
 $\langle \text{proof} \rangle$

**lemma** *starP-star-of* [*simp*]:  $( *p* P ) ( \text{star-of } x ) = P x$   
 $\langle \text{proof} \rangle$

**lemma** *starP2-star-of* [*simp*]:  $( *p2* P ) ( \text{star-of } x ) = *p* P x$   
 $\langle \text{proof} \rangle$

## 5.7 Internal sets

**definition**

*Iset* :: 'a set star  $\Rightarrow$  'a star set **where**  
*Iset* A =  $\{x. ( *p2* \text{op} \in ) x A\}$

**lemma** *Iset-star-n*:

$( \text{star-n } X \in \text{Iset } ( \text{star-n } A ) ) = ( \{n. X n \in A n\} \in \mathcal{U} )$   
 $\langle \text{proof} \rangle$

Transfer tactic should remove occurrences of *Iset*

$\langle \text{ML} \rangle$

**lemma** *transfer-mem* [*transfer-intro*]:

$\llbracket x \equiv \text{star-n } X; a \equiv \text{Iset } ( \text{star-n } A ) \rrbracket$   
 $\implies x \in a \equiv \{n. X n \in A n\} \in \mathcal{U}$   
 $\langle \text{proof} \rangle$

**lemma** *transfer-Collect* [*transfer-intro*]:

$\llbracket \bigwedge X. p ( \text{star-n } X ) \equiv \{n. P n (X n)\} \in \mathcal{U} \rrbracket$   
 $\implies \text{Collect } p \equiv \text{Iset } ( \text{star-n } ( \lambda n. \text{Collect } ( P n ) ) )$   
 $\langle \text{proof} \rangle$

**lemma** *transfer-set-eq* [*transfer-intro*]:

$\llbracket a \equiv \text{Iset } ( \text{star-n } A ); b \equiv \text{Iset } ( \text{star-n } B ) \rrbracket$   
 $\implies a = b \equiv \{n. A n = B n\} \in \mathcal{U}$   
 $\langle \text{proof} \rangle$

**lemma** *transfer-ball* [*transfer-intro*]:

$\llbracket a \equiv \text{Iset } ( \text{star-n } A ); \bigwedge X. p ( \text{star-n } X ) \equiv \{n. P n (X n)\} \in \mathcal{U} \rrbracket$   
 $\implies \forall x \in a. p x \equiv \{n. \forall x \in A n. P n x\} \in \mathcal{U}$   
 $\langle \text{proof} \rangle$

**lemma** *transfer-bea* [*transfer-intro*]:

$$\begin{aligned} & \llbracket a \equiv \text{Iset } (\text{star-}n \ A); \bigwedge X. p \ (\text{star-}n \ X) \equiv \{n. P \ n \ (X \ n)\} \in \mathcal{U} \rrbracket \\ & \implies \exists x \in a. p \ x \equiv \{n. \exists x \in A \ n. P \ n \ x\} \in \mathcal{U} \\ & \langle \text{proof} \rangle \end{aligned}$$

**lemma** *transfer-Iset* [*transfer-intro*]:

$$\begin{aligned} & \llbracket a \equiv \text{star-}n \ A \rrbracket \implies \text{Iset } a \equiv \text{Iset } (\text{star-}n \ (\lambda n. A \ n)) \\ & \langle \text{proof} \rangle \end{aligned}$$

Nonstandard extensions of sets.

**definition**

*starset* :: 'a set  $\Rightarrow$  'a star set (*\*s\** - [80] 80) **where**  
*starset* A = *Iset* (*star-of* A)

**declare** *starset-def* [*transfer-unfold*]

**lemma** *starset-mem*: (*star-of*  $x \in$  *\*s\** A) = ( $x \in$  A)  
 $\langle \text{proof} \rangle$

**lemma** *starset-UNIV*: *\*s\** (UNIV::'a set) = (UNIV::'a star set)  
 $\langle \text{proof} \rangle$

**lemma** *starset-empty*: *\*s\** {} = {}  
 $\langle \text{proof} \rangle$

**lemma** *starset-insert*: *\*s\** (*insert*  $x$  A) = *insert* (*star-of*  $x$ ) (*\*s\** A)  
 $\langle \text{proof} \rangle$

**lemma** *starset-Un*: *\*s\** (A  $\cup$  B) = *\*s\** A  $\cup$  *\*s\** B  
 $\langle \text{proof} \rangle$

**lemma** *starset-Int*: *\*s\** (A  $\cap$  B) = *\*s\** A  $\cap$  *\*s\** B  
 $\langle \text{proof} \rangle$

**lemma** *starset-Compl*: *\*s\**  $-$ A =  $-$ (*\*s\** A)  
 $\langle \text{proof} \rangle$

**lemma** *starset-diff*: *\*s\** (A  $-$  B) = *\*s\** A  $-$  *\*s\** B  
 $\langle \text{proof} \rangle$

**lemma** *starset-image*: *\*s\** ( $f$  ' A) = ( $*f*$   $f$ ) ' (*\*s\** A)  
 $\langle \text{proof} \rangle$

**lemma** *starset-vimage*: *\*s\** ( $f$   $-$ ' A) = ( $*f*$   $f$ )  $-$ ' (*\*s\** A)  
 $\langle \text{proof} \rangle$

**lemma** *starset-subset*: (*\*s\** A  $\subseteq$  *\*s\** B) = (A  $\subseteq$  B)  
 $\langle \text{proof} \rangle$

**lemma** *starset-eq*:  $(\text{*s* } A = \text{*s* } B) = (A = B)$   
 $\langle \textit{proof} \rangle$

**lemmas** *starset-simps* [*simp*] =  
*starset-mem*    *starset-UNIV*  
*starset-empty*    *starset-insert*  
*starset-Un*    *starset-Int*  
*starset-Compl*    *starset-diff*  
*starset-image*    *starset-vimage*  
*starset-subset*    *starset-eq*

## 5.8 Syntactic classes

**instantiation** *star* :: (*zero*) *zero*  
**begin**

**definition**

*star-zero-def*:  $0 \equiv \textit{star-of } 0$

**instance**  $\langle \textit{proof} \rangle$

**end**

**instantiation** *star* :: (*one*) *one*  
**begin**

**definition**

*star-one-def*:  $1 \equiv \textit{star-of } 1$

**instance**  $\langle \textit{proof} \rangle$

**end**

**instantiation** *star* :: (*plus*) *plus*  
**begin**

**definition**

*star-add-def*:  $(op +) \equiv \textit{*f2* } (op +)$

**instance**  $\langle \textit{proof} \rangle$

**end**

**instantiation** *star* :: (*times*) *times*  
**begin**

**definition**

*star-mult-def*:  $(op *) \equiv \textit{*f2* } (op *)$

```

instance ⟨proof⟩

end

instantiation star :: (uminus) uminus
begin

definition
  star-minus-def:  $uminus \equiv *f* \textit{uminus}$ 

instance ⟨proof⟩

end

instantiation star :: (minus) minus
begin

definition
  star-diff-def:  $(op \ -) \equiv *f2* (op \ -)$ 

instance ⟨proof⟩

end

instantiation star :: (abs) abs
begin

definition
  star-abs-def:  $abs \equiv *f* \textit{abs}$ 

instance ⟨proof⟩

end

instantiation star :: (sgn) sgn
begin

definition
  star-sgn-def:  $sgn \equiv *f* \textit{sgn}$ 

instance ⟨proof⟩

end

instantiation star :: (inverse) inverse
begin

definition

```

*star-divide-def*:  $(op /) \equiv *f2* (op /)$

**definition**

*star-inverse-def*:  $inverse \equiv *f* inverse$

**instance**  $\langle proof \rangle$

**end**

**instance** *star* :: (*Rings.dvd*) *Rings.dvd*  $\langle proof \rangle$

**instantiation** *star* :: (*Divides.div*) *Divides.div*  
**begin**

**definition**

*star-div-def*:  $(op div) \equiv *f2* (op div)$

**definition**

*star-mod-def*:  $(op mod) \equiv *f2* (op mod)$

**instance**  $\langle proof \rangle$

**end**

**instantiation** *star* :: (*ord*) *ord*  
**begin**

**definition**

*star-le-def*:  $(op \leq) \equiv *p2* (op \leq)$

**definition**

*star-less-def*:  $(op <) \equiv *p2* (op <)$

**instance**  $\langle proof \rangle$

**end**

**lemmas** *star-class-defs* [*transfer-unfold*] =  
*star-zero-def*    *star-one-def*  
*star-add-def*    *star-diff-def*    *star-minus-def*  
*star-mult-def*    *star-divide-def*    *star-inverse-def*  
*star-le-def*    *star-less-def*    *star-abs-def*    *star-sgn-def*  
*star-div-def*    *star-mod-def*

Class operations preserve standard elements

**lemma** *Standard-zero*:  $0 \in Standard$   
 $\langle proof \rangle$

**lemma** *Standard-one*:  $1 \in Standard$

*<proof>*

**lemma** *Standard-add*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x + y \in \text{Standard}$   
*<proof>*

**lemma** *Standard-diff*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x - y \in \text{Standard}$   
*<proof>*

**lemma** *Standard-minus*:  $x \in \text{Standard} \implies -x \in \text{Standard}$   
*<proof>*

**lemma** *Standard-mult*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x * y \in \text{Standard}$   
*<proof>*

**lemma** *Standard-divide*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x / y \in \text{Standard}$   
*<proof>*

**lemma** *Standard-inverse*:  $x \in \text{Standard} \implies \text{inverse } x \in \text{Standard}$   
*<proof>*

**lemma** *Standard-abs*:  $x \in \text{Standard} \implies \text{abs } x \in \text{Standard}$   
*<proof>*

**lemma** *Standard-div*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x \text{ div } y \in \text{Standard}$   
*<proof>*

**lemma** *Standard-mod*:  $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies x \text{ mod } y \in \text{Standard}$   
*<proof>*

**lemmas** *Standard-simps* [*simp*] =  
*Standard-zero Standard-one*  
*Standard-add Standard-diff Standard-minus*  
*Standard-mult Standard-divide Standard-inverse*  
*Standard-abs Standard-div Standard-mod*

*star-of* preserves class operations

**lemma** *star-of-add*:  $\text{star-of } (x + y) = \text{star-of } x + \text{star-of } y$   
*<proof>*

**lemma** *star-of-diff*:  $\text{star-of } (x - y) = \text{star-of } x - \text{star-of } y$   
*<proof>*

**lemma** *star-of-minus*:  $\text{star-of } (-x) = - \text{star-of } x$   
*<proof>*

**lemma** *star-of-mult*:  $\text{star-of } (x * y) = \text{star-of } x * \text{star-of } y$   
*<proof>*

**lemma** *star-of-divide*:  $\text{star-of } (x / y) = \text{star-of } x / \text{star-of } y$

*<proof>*

**lemma** *star-of-inverse*:  $\text{star-of } (\text{inverse } x) = \text{inverse } (\text{star-of } x)$   
*<proof>*

**lemma** *star-of-div*:  $\text{star-of } (x \text{ div } y) = \text{star-of } x \text{ div } \text{star-of } y$   
*<proof>*

**lemma** *star-of-mod*:  $\text{star-of } (x \text{ mod } y) = \text{star-of } x \text{ mod } \text{star-of } y$   
*<proof>*

**lemma** *star-of-abs*:  $\text{star-of } (\text{abs } x) = \text{abs } (\text{star-of } x)$   
*<proof>*

*star-of* preserves numerals

**lemma** *star-of-zero*:  $\text{star-of } 0 = 0$   
*<proof>*

**lemma** *star-of-one*:  $\text{star-of } 1 = 1$   
*<proof>*

*star-of* preserves orderings

**lemma** *star-of-less*:  $(\text{star-of } x < \text{star-of } y) = (x < y)$   
*<proof>*

**lemma** *star-of-le*:  $(\text{star-of } x \leq \text{star-of } y) = (x \leq y)$   
*<proof>*

**lemma** *star-of-eq*:  $(\text{star-of } x = \text{star-of } y) = (x = y)$   
*<proof>*

As above, for 0

**lemmas** *star-of-0-less* = *star-of-less* [*of 0, simplified star-of-zero*]

**lemmas** *star-of-0-le* = *star-of-le* [*of 0, simplified star-of-zero*]

**lemmas** *star-of-0-eq* = *star-of-eq* [*of 0, simplified star-of-zero*]

**lemmas** *star-of-less-0* = *star-of-less* [*of - 0, simplified star-of-zero*]

**lemmas** *star-of-le-0* = *star-of-le* [*of - 0, simplified star-of-zero*]

**lemmas** *star-of-eq-0* = *star-of-eq* [*of - 0, simplified star-of-zero*]

As above, for 1

**lemmas** *star-of-1-less* = *star-of-less* [*of 1, simplified star-of-one*]

**lemmas** *star-of-1-le* = *star-of-le* [*of 1, simplified star-of-one*]

**lemmas** *star-of-1-eq* = *star-of-eq* [*of 1, simplified star-of-one*]

**lemmas** *star-of-less-1* = *star-of-less* [*of - 1, simplified star-of-one*]

**lemmas** *star-of-le-1* = *star-of-le* [*of - 1, simplified star-of-one*]

**lemmas** *star-of-eq-1* = *star-of-eq* [*of - 1, simplified star-of-one*]

**lemmas** *star-of-simps* [*simp*] =  
*star-of-add*    *star-of-diff*    *star-of-minus*  
*star-of-mult*    *star-of-divide*    *star-of-inverse*  
*star-of-div*    *star-of-mod*    *star-of-abs*  
*star-of-zero*    *star-of-one*  
*star-of-less*    *star-of-le*    *star-of-eq*  
*star-of-0-less*    *star-of-0-le*    *star-of-0-eq*  
*star-of-less-0*    *star-of-le-0*    *star-of-eq-0*  
*star-of-1-less*    *star-of-1-le*    *star-of-1-eq*  
*star-of-less-1*    *star-of-le-1*    *star-of-eq-1*

## 5.9 Ordering and lattice classes

**instance** *star* :: (*order*) *order*  
 ⟨*proof*⟩

**instantiation** *star* :: (*semilattice-inf*) *semilattice-inf*  
**begin**

**definition**

*star-inf-def* [*transfer-unfold*]:  $inf \equiv *f2* inf$

**instance**  
 ⟨*proof*⟩

**end**

**instantiation** *star* :: (*semilattice-sup*) *semilattice-sup*  
**begin**

**definition**

*star-sup-def* [*transfer-unfold*]:  $sup \equiv *f2* sup$

**instance**  
 ⟨*proof*⟩

**end**

**instance** *star* :: (*lattice*) *lattice* ⟨*proof*⟩

**instance** *star* :: (*distrib-lattice*) *distrib-lattice*  
 ⟨*proof*⟩

**lemma** *Standard-inf* [*simp*]:

$\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies inf\ x\ y \in \text{Standard}$   
 ⟨*proof*⟩

**lemma** *Standard-sup* [*simp*]:

$\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies \text{sup } x \ y \in \text{Standard}$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-inf* [simp]:  $\text{star-of } (\text{inf } x \ y) = \text{inf } (\text{star-of } x) (\text{star-of } y)$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-sup* [simp]:  $\text{star-of } (\text{sup } x \ y) = \text{sup } (\text{star-of } x) (\text{star-of } y)$   
 $\langle \text{proof} \rangle$

**instance** *star* :: (linorder) linorder  
 $\langle \text{proof} \rangle$

**lemma** *star-max-def* [transfer-unfold]:  $\text{max} = *f2* \ \text{max}$   
 $\langle \text{proof} \rangle$

**lemma** *star-min-def* [transfer-unfold]:  $\text{min} = *f2* \ \text{min}$   
 $\langle \text{proof} \rangle$

**lemma** *Standard-max* [simp]:  
 $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies \text{max } x \ y \in \text{Standard}$   
 $\langle \text{proof} \rangle$

**lemma** *Standard-min* [simp]:  
 $\llbracket x \in \text{Standard}; y \in \text{Standard} \rrbracket \implies \text{min } x \ y \in \text{Standard}$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-max* [simp]:  $\text{star-of } (\text{max } x \ y) = \text{max } (\text{star-of } x) (\text{star-of } y)$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-min* [simp]:  $\text{star-of } (\text{min } x \ y) = \text{min } (\text{star-of } x) (\text{star-of } y)$   
 $\langle \text{proof} \rangle$

## 5.10 Ordered group classes

**instance** *star* :: (semigroup-add) semigroup-add  
 $\langle \text{proof} \rangle$

**instance** *star* :: (ab-semigroup-add) ab-semigroup-add  
 $\langle \text{proof} \rangle$

**instance** *star* :: (semigroup-mult) semigroup-mult  
 $\langle \text{proof} \rangle$

**instance** *star* :: (ab-semigroup-mult) ab-semigroup-mult  
 $\langle \text{proof} \rangle$

**instance** *star* :: (comm-monoid-add) comm-monoid-add  
 $\langle \text{proof} \rangle$

**instance** *star* :: (*monoid-mult*) *monoid-mult*  
 ⟨*proof*⟩

**instance** *star* :: (*comm-monoid-mult*) *comm-monoid-mult*  
 ⟨*proof*⟩

**instance** *star* :: (*cancel-semigroup-add*) *cancel-semigroup-add*  
 ⟨*proof*⟩

**instance** *star* :: (*cancel-ab-semigroup-add*) *cancel-ab-semigroup-add*  
 ⟨*proof*⟩

**instance** *star* :: (*cancel-comm-monoid-add*) *cancel-comm-monoid-add* ⟨*proof*⟩

**instance** *star* :: (*ab-group-add*) *ab-group-add*  
 ⟨*proof*⟩

**instance** *star* :: (*ordered-ab-semigroup-add*) *ordered-ab-semigroup-add*  
 ⟨*proof*⟩

**instance** *star* :: (*ordered-cancel-ab-semigroup-add*) *ordered-cancel-ab-semigroup-add*  
 ⟨*proof*⟩

**instance** *star* :: (*ordered-ab-semigroup-add-imp-le*) *ordered-ab-semigroup-add-imp-le*  
 ⟨*proof*⟩

**instance** *star* :: (*ordered-comm-monoid-add*) *ordered-comm-monoid-add* ⟨*proof*⟩

**instance** *star* :: (*ordered-ab-group-add*) *ordered-ab-group-add* ⟨*proof*⟩

**instance** *star* :: (*ordered-ab-group-add-abs*) *ordered-ab-group-add-abs*  
 ⟨*proof*⟩

**instance** *star* :: (*linordered-cancel-ab-semigroup-add*) *linordered-cancel-ab-semigroup-add*  
 ⟨*proof*⟩

## 5.11 Ring and field classes

**instance** *star* :: (*semiring*) *semiring*  
 ⟨*proof*⟩

**instance** *star* :: (*semiring-0*) *semiring-0*  
 ⟨*proof*⟩

**instance** *star* :: (*semiring-0-cancel*) *semiring-0-cancel* ⟨*proof*⟩

**instance** *star* :: (*comm-semiring*) *comm-semiring*  
 ⟨*proof*⟩

**instance** *star* :: (*comm-semiring-0*) *comm-semiring-0* ⟨*proof*⟩

```

instance star :: (comm-semiring-0-cancel) comm-semiring-0-cancel ⟨proof⟩

instance star :: (zero-neq-one) zero-neq-one
⟨proof⟩

instance star :: (semiring-1) semiring-1 ⟨proof⟩
instance star :: (comm-semiring-1) comm-semiring-1 ⟨proof⟩

instance star :: (no-zero-divisors) no-zero-divisors
⟨proof⟩

instance star :: (semiring-1-cancel) semiring-1-cancel ⟨proof⟩
instance star :: (comm-semiring-1-cancel) comm-semiring-1-cancel ⟨proof⟩
instance star :: (ring) ring ⟨proof⟩
instance star :: (comm-ring) comm-ring ⟨proof⟩
instance star :: (ring-1) ring-1 ⟨proof⟩
instance star :: (comm-ring-1) comm-ring-1 ⟨proof⟩
instance star :: (ring-no-zero-divisors) ring-no-zero-divisors ⟨proof⟩
instance star :: (ring-1-no-zero-divisors) ring-1-no-zero-divisors ⟨proof⟩
instance star :: (idom) idom ⟨proof⟩

instance star :: (division-ring) division-ring
⟨proof⟩

instance star :: (division-ring-inverse-zero) division-ring-inverse-zero
⟨proof⟩

instance star :: (field) field
⟨proof⟩

instance star :: (field-inverse-zero) field-inverse-zero
⟨proof⟩

instance star :: (ordered-semiring) ordered-semiring
⟨proof⟩

instance star :: (ordered-cancel-semiring) ordered-cancel-semiring ⟨proof⟩

instance star :: (linordered-semiring-strict) linordered-semiring-strict
⟨proof⟩

instance star :: (ordered-comm-semiring) ordered-comm-semiring
⟨proof⟩

instance star :: (ordered-cancel-comm-semiring) ordered-cancel-comm-semiring ⟨proof⟩

instance star :: (linordered-comm-semiring-strict) linordered-comm-semiring-strict
⟨proof⟩

```

**instance** *star* :: (*ordered-ring*) *ordered-ring* ⟨*proof*⟩

**instance** *star* :: (*ordered-ring-abs*) *ordered-ring-abs*  
 ⟨*proof*⟩

**instance** *star* :: (*abs-if*) *abs-if*  
 ⟨*proof*⟩

**instance** *star* :: (*sgn-if*) *sgn-if*  
 ⟨*proof*⟩

**instance** *star* :: (*linordered-ring-strict*) *linordered-ring-strict* ⟨*proof*⟩

**instance** *star* :: (*ordered-comm-ring*) *ordered-comm-ring* ⟨*proof*⟩

**instance** *star* :: (*linordered-semidom*) *linordered-semidom*  
 ⟨*proof*⟩

**instance** *star* :: (*linordered-idom*) *linordered-idom* ⟨*proof*⟩

**instance** *star* :: (*linordered-field*) *linordered-field* ⟨*proof*⟩

**instance** *star* :: (*linordered-field-inverse-zero*) *linordered-field-inverse-zero* ⟨*proof*⟩

## 5.12 Power

**lemma** *star-power-def* [*transfer-unfold*]:

$(op \ ^) \equiv \lambda x n. ( *f* (\lambda x. x \ ^ n) ) x$   
 ⟨*proof*⟩

**lemma** *Standard-power* [*simp*]:  $x \in \text{Standard} \implies x \ ^ n \in \text{Standard}$   
 ⟨*proof*⟩

**lemma** *star-of-power* [*simp*]:  $\text{star-of } (x \ ^ n) = \text{star-of } x \ ^ n$   
 ⟨*proof*⟩

## 5.13 Number classes

**instance** *star* :: (*numeral*) *numeral* ⟨*proof*⟩

**lemma** *star-numeral-def* [*transfer-unfold*]:

$\text{numeral } k = \text{star-of } (\text{numeral } k)$   
 ⟨*proof*⟩

**lemma** *Standard-numeral* [*simp*]:  $\text{numeral } k \in \text{Standard}$   
 ⟨*proof*⟩

**lemma** *star-of-numeral* [*simp*]:  $\text{star-of } (\text{numeral } k) = \text{numeral } k$   
 ⟨*proof*⟩

**lemma** *star-neg-numeral-def* [*transfer-unfold*]:

$\text{neg-numeral } k = \text{star-of } (\text{neg-numeral } k)$   
 ⟨*proof*⟩

**lemma** *Standard-neg-numeral* [simp]: *neg-numeral*  $k \in \text{Standard}$   
 ⟨proof⟩

**lemma** *star-of-neg-numeral* [simp]: *star-of* (*neg-numeral*  $k$ ) = *neg-numeral*  $k$   
 ⟨proof⟩

**lemma** *star-of-nat-def* [transfer-unfold]: *of-nat*  $n = \text{star-of}$  (*of-nat*  $n$ )  
 ⟨proof⟩

**lemmas** *star-of-compare-numeral* [simp] =  
*star-of-less* [of numeral  $k$ , simplified *star-of-numeral*]  
*star-of-le* [of numeral  $k$ , simplified *star-of-numeral*]  
*star-of-eq* [of numeral  $k$ , simplified *star-of-numeral*]  
*star-of-less* [of - numeral  $k$ , simplified *star-of-numeral*]  
*star-of-le* [of - numeral  $k$ , simplified *star-of-numeral*]  
*star-of-eq* [of - numeral  $k$ , simplified *star-of-numeral*]  
*star-of-less* [of neg-numeral  $k$ , simplified *star-of-numeral*]  
*star-of-le* [of neg-numeral  $k$ , simplified *star-of-numeral*]  
*star-of-eq* [of neg-numeral  $k$ , simplified *star-of-numeral*]  
*star-of-less* [of - neg-numeral  $k$ , simplified *star-of-numeral*]  
*star-of-le* [of - neg-numeral  $k$ , simplified *star-of-numeral*]  
*star-of-eq* [of - neg-numeral  $k$ , simplified *star-of-numeral*] **for**  $k$

**lemma** *Standard-of-nat* [simp]: *of-nat*  $n \in \text{Standard}$   
 ⟨proof⟩

**lemma** *star-of-of-nat* [simp]: *star-of* (*of-nat*  $n$ ) = *of-nat*  $n$   
 ⟨proof⟩

**lemma** *star-of-int-def* [transfer-unfold]: *of-int*  $z = \text{star-of}$  (*of-int*  $z$ )  
 ⟨proof⟩

**lemma** *Standard-of-int* [simp]: *of-int*  $z \in \text{Standard}$   
 ⟨proof⟩

**lemma** *star-of-of-int* [simp]: *star-of* (*of-int*  $z$ ) = *of-int*  $z$   
 ⟨proof⟩

**instance** *star* :: (*semiring-char-0*) *semiring-char-0* ⟨proof⟩

**instance** *star* :: (*ring-char-0*) *ring-char-0* ⟨proof⟩

## 5.14 Finite class

**lemma** *starset-finite*: *finite*  $A \implies *s* A = \text{star-of} ' A$   
 ⟨proof⟩

**instance** *star* :: (*finite*) *finite*  
 ⟨proof⟩

end

## 6 HyperNat: Hypernatural numbers

**theory** *HyperNat*  
**imports** *StarDef*  
**begin**

**type-synonym** *hypnat* = *nat star*

**abbreviation**

*hypnat-of-nat* :: *nat* => *nat star* **where**  
*hypnat-of-nat* == *star-of*

**definition**

*hSuc* :: *hypnat* => *hypnat* **where**  
*hSuc-def* [*transfer-unfold*]: *hSuc* = *\*f\** *Suc*

### 6.1 Properties Transferred from Naturals

**lemma** *hSuc-not-zero* [*iff*]:  $\bigwedge m. hSuc\ m \neq 0$   
 ⟨*proof*⟩

**lemma** *zero-not-hSuc* [*iff*]:  $\bigwedge m. 0 \neq hSuc\ m$   
 ⟨*proof*⟩

**lemma** *hSuc-hSuc-eq* [*iff*]:  $\bigwedge m\ n. (hSuc\ m = hSuc\ n) = (m = n)$   
 ⟨*proof*⟩

**lemma** *zero-less-hSuc* [*iff*]:  $\bigwedge n. 0 < hSuc\ n$   
 ⟨*proof*⟩

**lemma** *hypnat-minus-zero* [*simp*]:  $!!z. z - z = (0::hypnat)$   
 ⟨*proof*⟩

**lemma** *hypnat-diff-0-eq-0* [*simp*]:  $!!n. (0::hypnat) - n = 0$   
 ⟨*proof*⟩

**lemma** *hypnat-add-is-0* [*iff*]:  $!!m\ n. (m+n = (0::hypnat)) = (m=0 \ \&\ n=0)$   
 ⟨*proof*⟩

**lemma** *hypnat-diff-diff-left*:  $!!i\ j\ k. (i::hypnat) - j - k = i - (j+k)$   
 ⟨*proof*⟩

**lemma** *hypnat-diff-commute*:  $!!i\ j\ k. (i::hypnat) - j - k = i - k - j$   
 ⟨*proof*⟩

**lemma** *hypnat-diff-add-inverse* [simp]:  $!!m n. ((n::hypnat) + m) - n = m$   
 ⟨proof⟩

**lemma** *hypnat-diff-add-inverse2* [simp]:  $!!m n. ((m::hypnat) + n) - n = m$   
 ⟨proof⟩

**lemma** *hypnat-diff-cancel* [simp]:  $!!k m n. ((k::hypnat) + m) - (k+n) = m - n$   
 ⟨proof⟩

**lemma** *hypnat-diff-cancel2* [simp]:  $!!k m n. ((m::hypnat) + k) - (n+k) = m - n$   
 ⟨proof⟩

**lemma** *hypnat-diff-add-0* [simp]:  $!!m n. (n::hypnat) - (n+m) = (0::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-diff-mult-distrib*:  $!!k m n. ((m::hypnat) - n) * k = (m * k) - (n * k)$   
 ⟨proof⟩

**lemma** *hypnat-diff-mult-distrib2*:  $!!k m n. (k::hypnat) * (m - n) = (k * m) - (k * n)$   
 ⟨proof⟩

**lemma** *hypnat-le-zero-cancel* [iff]:  $!!n. (n \leq (0::hypnat)) = (n = 0)$   
 ⟨proof⟩

**lemma** *hypnat-mult-is-0* [simp]:  $!!m n. (m*n = (0::hypnat)) = (m=0 \mid n=0)$   
 ⟨proof⟩

**lemma** *hypnat-diff-is-0-eq* [simp]:  $!!m n. ((m::hypnat) - n = 0) = (m \leq n)$   
 ⟨proof⟩

**lemma** *hypnat-not-less0* [iff]:  $!!n. \sim n < (0::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-less-one* [iff]:  
 $!!n. (n < (1::hypnat)) = (n=0)$   
 ⟨proof⟩

**lemma** *hypnat-add-diff-inverse*:  $!!m n. \sim m < n ==> n+(m-n) = (m::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-le-add-diff-inverse* [simp]:  $!!m n. n \leq m ==> n+(m-n) = (m::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-le-add-diff-inverse2* [simp]:  $!!m n. n \leq m ==> (m-n)+n = (m::hypnat)$   
 ⟨proof⟩

**declare** *hypnat-le-add-diff-inverse2* [OF order-less-imp-le]

**lemma** *hypnat-le0* [iff]:  $!!n. (0::hypnat) \leq n$   
 ⟨proof⟩

**lemma** *hypnat-le-add1* [simp]:  $!!x n. (x::hypnat) \leq x + n$   
 ⟨proof⟩

**lemma** *hypnat-add-self-le* [simp]:  $!!x n. (x::hypnat) \leq n + x$   
 ⟨proof⟩

**lemma** *hypnat-add-one-self-less* [simp]:  $(x::hypnat) < x + (1::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-neq0-conv* [iff]:  $!!n. (n \neq 0) = (0 < (n::hypnat))$   
 ⟨proof⟩

**lemma** *hypnat-gt-zero-iff*:  $((0::hypnat) < n) = ((1::hypnat) \leq n)$   
 ⟨proof⟩

**lemma** *hypnat-gt-zero-iff2*:  $(0 < n) = (\exists m. n = m + (1::hypnat))$   
 ⟨proof⟩

**lemma** *hypnat-add-self-not-less*:  $\sim (x + y < (x::hypnat))$   
 ⟨proof⟩

**lemma** *hypnat-diff-split*:  
 $P(a - b::hypnat) = ((a < b \longrightarrow P 0) \& (ALL d. a = b + d \longrightarrow P d))$   
 — elimination of  $-$  on *hypnat*  
 ⟨proof⟩

## 6.2 Properties of the set of embedded natural numbers

**lemma** *of-nat-eq-star-of* [simp]: *of-nat* = *star-of*  
 ⟨proof⟩

**lemma** *Nats-eq-Standard*:  $(Nats :: nat \text{ star set}) = \text{Standard}$   
 ⟨proof⟩

**lemma** *hypnat-of-nat-mem-Nats* [simp]: *hypnat-of-nat*  $n \in Nats$   
 ⟨proof⟩

**lemma** *hypnat-of-nat-one* [simp]: *hypnat-of-nat*  $(\text{Suc } 0) = (1::hypnat)$   
 ⟨proof⟩

**lemma** *hypnat-of-nat-Suc* [simp]:  
 $\text{hypnat-of-nat } (\text{Suc } n) = \text{hypnat-of-nat } n + (1::hypnat)$   
 ⟨proof⟩

**lemma** *of-nat-eq-add* [rule-format]:

$\forall d::\text{hypnat. of-nat } m = \text{of-nat } n + d \dashv\vdash d \in \text{range of-nat}$   
 ⟨proof⟩

**lemma** *Nats-diff* [simp]:  $[[a \in \text{Nats}; b \in \text{Nats}] \implies (a-b :: \text{hypnat}) \in \text{Nats}$   
 ⟨proof⟩

### 6.3 Infinite Hypernatural Numbers – *HNatInfinite*

**definition**

*HNatInfinite* :: hypnat set **where**  
*HNatInfinite* = {*n*. *n* ∉ *Nats*}

**lemma** *Nats-not-HNatInfinite-iff*:  $(x \in \text{Nats}) = (x \notin \text{HNatInfinite})$   
 ⟨proof⟩

**lemma** *HNatInfinite-not-Nats-iff*:  $(x \in \text{HNatInfinite}) = (x \notin \text{Nats})$   
 ⟨proof⟩

**lemma** *star-of-neq-HNatInfinite*:  $N \in \text{HNatInfinite} \implies \text{star-of } n \neq N$   
 ⟨proof⟩

**lemma** *star-of-Suc-lessI*:  
 $\bigwedge N. [[\text{star-of } n < N; \text{star-of } (\text{Suc } n) \neq N] \implies \text{star-of } (\text{Suc } n) < N$   
 ⟨proof⟩

**lemma** *star-of-less-HNatInfinite*:  
**assumes** *N*:  $N \in \text{HNatInfinite}$   
**shows** *star-of* *n* < *N*  
 ⟨proof⟩

**lemma** *star-of-le-HNatInfinite*:  $N \in \text{HNatInfinite} \implies \text{star-of } n \leq N$   
 ⟨proof⟩

#### 6.3.1 Closure Rules

**lemma** *Nats-less-HNatInfinite*:  $[[x \in \text{Nats}; y \in \text{HNatInfinite}] \implies x < y$   
 ⟨proof⟩

**lemma** *Nats-le-HNatInfinite*:  $[[x \in \text{Nats}; y \in \text{HNatInfinite}] \implies x \leq y$   
 ⟨proof⟩

**lemma** *zero-less-HNatInfinite*:  $x \in \text{HNatInfinite} \implies 0 < x$   
 ⟨proof⟩

**lemma** *one-less-HNatInfinite*:  $x \in \text{HNatInfinite} \implies 1 < x$   
 ⟨proof⟩

**lemma** *one-le-HNatInfinite*:  $x \in \text{HNatInfinite} \implies 1 \leq x$   
 ⟨proof⟩

**lemma** *zero-not-mem-HNatInfinite* [simp]:  $0 \notin \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *Nats-downward-closed*:  
 $\llbracket x \in \text{Nats}; (y::\text{hypnat}) \leq x \rrbracket \implies y \in \text{Nats}$   
 ⟨proof⟩

**lemma** *HNatInfinite-upward-closed*:  
 $\llbracket x \in \text{HNatInfinite}; x \leq y \rrbracket \implies y \in \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *HNatInfinite-add*:  $x \in \text{HNatInfinite} \implies x + y \in \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *HNatInfinite-add-one*:  $x \in \text{HNatInfinite} \implies x + 1 \in \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *HNatInfinite-diff*:  
 $\llbracket x \in \text{HNatInfinite}; y \in \text{Nats} \rrbracket \implies x - y \in \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *HNatInfinite-is-Suc*:  $x \in \text{HNatInfinite} \implies \exists y. x = y + (1::\text{hypnat})$   
 ⟨proof⟩

## 6.4 Existence of an infinite hypernatural number

### definition

*whn* :: hypnat **where**  
*hypnat-omega-def*:  $\text{whn} = \text{star-}n \ (\%n::\text{nat. } n)$

**lemma** *hypnat-of-nat-neq-whn*:  $\text{hypnat-of-nat } n \neq \text{whn}$   
 ⟨proof⟩

**lemma** *whn-neq-hypnat-of-nat*:  $\text{whn} \neq \text{hypnat-of-nat } n$   
 ⟨proof⟩

**lemma** *whn-not-Nats* [simp]:  $\text{whn} \notin \text{Nats}$   
 ⟨proof⟩

**lemma** *HNatInfinite-whn* [simp]:  $\text{whn} \in \text{HNatInfinite}$   
 ⟨proof⟩

**lemma** *lemma-unbounded-set* [simp]:  $\{n::\text{nat. } m < n\} \in \text{FreeUltrafilterNat}$   
 ⟨proof⟩

**lemma** *Compl-Collect-le*:  $\neg \{n::\text{nat. } N \leq n\} = \{n. n < N\}$   
 ⟨proof⟩

**lemma** *hypnat-of-nat-eq*:

$hypnat\text{-of-nat } m = star\text{-}n (\%n::nat. m)$   
 $\langle proof \rangle$

**lemma** *SHNat-eq*:  $Nats = \{n. \exists N. n = hypnat\text{-of-nat } N\}$

$\langle proof \rangle$

**lemma** *Nats-less-whn*:  $n \in Nats \implies n < whn$

$\langle proof \rangle$

**lemma** *Nats-le-whn*:  $n \in Nats \implies n \leq whn$

$\langle proof \rangle$

**lemma** *hypnat-of-nat-less-whn* [*simp*]:  $hypnat\text{-of-nat } n < whn$

$\langle proof \rangle$

**lemma** *hypnat-of-nat-le-whn* [*simp*]:  $hypnat\text{-of-nat } n \leq whn$

$\langle proof \rangle$

**lemma** *hypnat-zero-less-hypnat-omega* [*simp*]:  $0 < whn$

$\langle proof \rangle$

**lemma** *hypnat-one-less-hypnat-omega* [*simp*]:  $1 < whn$

$\langle proof \rangle$

#### 6.4.1 Alternative characterization of the set of infinite hypernaturals

$HNatInfinite = \{N. \forall n \in \mathbb{N}. n < N\}$

**lemma** *HNatInfinite-FreeUltrafilterNat-lemma*:

**assumes**  $\forall N::nat. \{n. f\ n \neq N\} \in FreeUltrafilterNat$

**shows**  $\{n. N < f\ n\} \in FreeUltrafilterNat$

$\langle proof \rangle$

**lemma** *HNatInfinite-iff*:  $HNatInfinite = \{N. \forall n \in Nats. n < N\}$

$\langle proof \rangle$

#### 6.4.2 Alternative Characterization of $HNatInfinite$ using Free Ultrafilter

**lemma** *HNatInfinite-FreeUltrafilterNat*:

$star\text{-}n\ X \in HNatInfinite \implies \forall u. \{n. u < X\ n\} \in FreeUltrafilterNat$

$\langle proof \rangle$

**lemma** *FreeUltrafilterNat-HNatInfinite*:

$\forall u. \{n. u < X\ n\} \in FreeUltrafilterNat \implies star\text{-}n\ X \in HNatInfinite$

$\langle proof \rangle$

**lemma** *HNatInfinite-FreeUltrafilterNat-iff*:

$(\text{star-}n \ X \in \text{HNatInfinite}) = (\forall u. \{n. u < X \ n\}: \text{FreeUltrafilterNat})$   
 ⟨proof⟩

## 6.5 Embedding of the Hypernaturals into other types

**definition**

*of-hypnat* :: *hypnat*  $\Rightarrow$  'a::semiring-1-cancel star **where**  
*of-hypnat-def* [*transfer-unfold*]: *of-hypnat* = \*f\* *of-nat*

**lemma** *of-hypnat-0* [*simp*]: *of-hypnat* 0 = 0  
 ⟨proof⟩

**lemma** *of-hypnat-1* [*simp*]: *of-hypnat* 1 = 1  
 ⟨proof⟩

**lemma** *of-hypnat-hSuc*:  $\bigwedge m. \text{of-hypnat} (\text{hSuc } m) = 1 + \text{of-hypnat } m$   
 ⟨proof⟩

**lemma** *of-hypnat-add* [*simp*]:  
 $\bigwedge m \ n. \text{of-hypnat} (m + n) = \text{of-hypnat } m + \text{of-hypnat } n$   
 ⟨proof⟩

**lemma** *of-hypnat-mult* [*simp*]:  
 $\bigwedge m \ n. \text{of-hypnat} (m * n) = \text{of-hypnat } m * \text{of-hypnat } n$   
 ⟨proof⟩

**lemma** *of-hypnat-less-iff* [*simp*]:  
 $\bigwedge m \ n. (\text{of-hypnat } m < (\text{of-hypnat } n::'a::\text{linordered-semidom star})) = (m < n)$   
 ⟨proof⟩

**lemma** *of-hypnat-0-less-iff* [*simp*]:  
 $\bigwedge n. (0 < (\text{of-hypnat } n::'a::\text{linordered-semidom star})) = (0 < n)$   
 ⟨proof⟩

**lemma** *of-hypnat-less-0-iff* [*simp*]:  
 $\bigwedge m. \neg (\text{of-hypnat } m::'a::\text{linordered-semidom star}) < 0$   
 ⟨proof⟩

**lemma** *of-hypnat-le-iff* [*simp*]:  
 $\bigwedge m \ n. (\text{of-hypnat } m \leq (\text{of-hypnat } n::'a::\text{linordered-semidom star})) = (m \leq n)$   
 ⟨proof⟩

**lemma** *of-hypnat-0-le-iff* [*simp*]:  
 $\bigwedge n. 0 \leq (\text{of-hypnat } n::'a::\text{linordered-semidom star})$   
 ⟨proof⟩

**lemma** *of-hypnat-le-0-iff* [*simp*]:  
 $\bigwedge m. ((\text{of-hypnat } m::'a::\text{linordered-semidom star}) \leq 0) = (m = 0)$

*<proof>*

**lemma** *of-hypnat-eq-iff [simp]*:

$\bigwedge m n. (\text{of-hypnat } m = (\text{of-hypnat } n :: 'a :: \text{linordered-semidom star})) = (m = n)$   
*<proof>*

**lemma** *of-hypnat-eq-0-iff [simp]*:

$\bigwedge m. ((\text{of-hypnat } m :: 'a :: \text{linordered-semidom star}) = 0) = (m = 0)$   
*<proof>*

**lemma** *HNatInfinite-of-hypnat-gt-zero*:

$N \in \text{HNatInfinite} \implies (0 :: 'a :: \text{linordered-semidom star}) < \text{of-hypnat } N$   
*<proof>*

**end**

## 7 HyperDef: Construction of Hyperreals Using Ultrafilters

**theory** *HyperDef*

**imports** *HyperNat Real*

**begin**

**type-synonym** *hypreal = real star*

**abbreviation**

*hypreal-of-real* :: *real => real star* **where**  
*hypreal-of-real* == *star-of*

**abbreviation**

*hypreal-of-hypnat* :: *hypnat => hypreal* **where**  
*hypreal-of-hypnat*  $\equiv$  *of-hypnat*

**definition**

*omega* :: *hypreal* **where**  
 — an infinite number = [*1, 2, 3, ...*]  
*omega* = *star-n* ( $\lambda n. \text{real } (\text{Suc } n)$ )

**definition**

*epsilon* :: *hypreal* **where**  
 — an infinitesimal number = [*1, 1/2, 1/3, ...*]  
*epsilon* = *star-n* ( $\lambda n. \text{inverse } (\text{real } (\text{Suc } n))$ )

**notation** (*xsymbols*)

*omega* ( $\omega$ ) **and**  
*epsilon* ( $\epsilon$ )

**notation** (*HTML output*)  
*omega* ( $\omega$ ) **and**  
*epsilon* ( $\varepsilon$ )

## 7.1 Real vector class instances

**instantiation** *star* :: (*scaleR*) *scaleR*  
**begin**

**definition**  
*star-scaleR-def* [*transfer-unfold*]: *scaleR* *r*  $\equiv$  *\*f\** (*scaleR* *r*)

**instance**  $\langle$ *proof* $\rangle$

**end**

**lemma** *Standard-scaleR* [*simp*]:  $x \in \text{Standard} \implies \text{scaleR } r \ x \in \text{Standard}$   
 $\langle$ *proof* $\rangle$

**lemma** *star-of-scaleR* [*simp*]:  $\text{star-of } (\text{scaleR } r \ x) = \text{scaleR } r \ (\text{star-of } x)$   
 $\langle$ *proof* $\rangle$

**instance** *star* :: (*real-vector*) *real-vector*  
 $\langle$ *proof* $\rangle$

**instance** *star* :: (*real-algebra*) *real-algebra*  
 $\langle$ *proof* $\rangle$

**instance** *star* :: (*real-algebra-1*) *real-algebra-1*  $\langle$ *proof* $\rangle$

**instance** *star* :: (*real-div-algebra*) *real-div-algebra*  $\langle$ *proof* $\rangle$

**instance** *star* :: (*field-char-0*) *field-char-0*  $\langle$ *proof* $\rangle$

**instance** *star* :: (*real-field*) *real-field*  $\langle$ *proof* $\rangle$

**lemma** *star-of-real-def* [*transfer-unfold*]:  $\text{of-real } r = \text{star-of } (\text{of-real } r)$   
 $\langle$ *proof* $\rangle$

**lemma** *Standard-of-real* [*simp*]:  $\text{of-real } r \in \text{Standard}$   
 $\langle$ *proof* $\rangle$

**lemma** *star-of-of-real* [*simp*]:  $\text{star-of } (\text{of-real } r) = \text{of-real } r$   
 $\langle$ *proof* $\rangle$

**lemma** *of-real-eq-star-of* [*simp*]:  $\text{of-real} = \text{star-of}$   
 $\langle$ *proof* $\rangle$

**lemma** *Reals-eq-Standard*:  $(\text{Reals} :: \text{hypreal set}) = \text{Standard}$

⟨proof⟩

## 7.2 Injection from *hypreal*

### definition

*of-hypreal* :: *hypreal*  $\Rightarrow$  'a::real-algebra-1 star **where**  
 [transfer-unfold]: *of-hypreal* = \*f\* *of-real*

**lemma** *Standard-of-hypreal* [simp]:

$r \in \text{Standard} \Longrightarrow \text{of-hypreal } r \in \text{Standard}$

⟨proof⟩

**lemma** *of-hypreal-0* [simp]: *of-hypreal* 0 = 0

⟨proof⟩

**lemma** *of-hypreal-1* [simp]: *of-hypreal* 1 = 1

⟨proof⟩

**lemma** *of-hypreal-add* [simp]:

$\bigwedge x y. \text{of-hypreal } (x + y) = \text{of-hypreal } x + \text{of-hypreal } y$

⟨proof⟩

**lemma** *of-hypreal-minus* [simp]:  $\bigwedge x. \text{of-hypreal } (-x) = - \text{of-hypreal } x$

⟨proof⟩

**lemma** *of-hypreal-diff* [simp]:

$\bigwedge x y. \text{of-hypreal } (x - y) = \text{of-hypreal } x - \text{of-hypreal } y$

⟨proof⟩

**lemma** *of-hypreal-mult* [simp]:

$\bigwedge x y. \text{of-hypreal } (x * y) = \text{of-hypreal } x * \text{of-hypreal } y$

⟨proof⟩

**lemma** *of-hypreal-inverse* [simp]:

$\bigwedge x. \text{of-hypreal } (\text{inverse } x) =$

$\text{inverse } (\text{of-hypreal } x :: 'a::\{\text{real-div-algebra, division-ring-inverse-zero}\} \text{ star})$

⟨proof⟩

**lemma** *of-hypreal-divide* [simp]:

$\bigwedge x y. \text{of-hypreal } (x / y) =$

$(\text{of-hypreal } x / \text{of-hypreal } y :: 'a::\{\text{real-field, field-inverse-zero}\} \text{ star})$

⟨proof⟩

**lemma** *of-hypreal-eq-iff* [simp]:

$\bigwedge x y. (\text{of-hypreal } x = \text{of-hypreal } y) = (x = y)$

⟨proof⟩

**lemma** *of-hypreal-eq-0-iff* [simp]:

$\bigwedge x. (\text{of-hypreal } x = 0) = (x = 0)$

⟨proof⟩

### 7.3 Properties of *starrel*

**lemma** *lemma-starrel-refl* [*simp*]:  $x \in \text{starrel} \{x\}$   
 ⟨proof⟩

**lemma** *starrel-in-hypreal* [*simp*]:  $\text{starrel} \{x\} : \text{star}$   
 ⟨proof⟩

**declare** *Abs-star-inject* [*simp*] *Abs-star-inverse* [*simp*]  
**declare** *equiv-starrel* [*THEN eq-equiv-class-iff, simp*]

### 7.4 *hypreal-of-real*: the Injection from *real* to *hypreal*

**lemma** *inj-star-of*: *inj star-of*  
 ⟨proof⟩

**lemma** *mem-Rep-star-iff*:  $(X \in \text{Rep-star } x) = (x = \text{star-n } X)$   
 ⟨proof⟩

**lemma** *Rep-star-star-n-iff* [*simp*]:  
 $(X \in \text{Rep-star } (\text{star-n } Y)) = (\{n. Y n = X n\} \in \mathcal{U})$   
 ⟨proof⟩

**lemma** *Rep-star-star-n*:  $X \in \text{Rep-star } (\text{star-n } X)$   
 ⟨proof⟩

### 7.5 Properties of *star-n*

**lemma** *star-n-add*:  
 $\text{star-n } X + \text{star-n } Y = \text{star-n } (\%n. X n + Y n)$   
 ⟨proof⟩

**lemma** *star-n-minus*:  
 $-\text{star-n } X = \text{star-n } (\%n. -(X n))$   
 ⟨proof⟩

**lemma** *star-n-diff*:  
 $\text{star-n } X - \text{star-n } Y = \text{star-n } (\%n. X n - Y n)$   
 ⟨proof⟩

**lemma** *star-n-mult*:  
 $\text{star-n } X * \text{star-n } Y = \text{star-n } (\%n. X n * Y n)$   
 ⟨proof⟩

**lemma** *star-n-inverse*:  
 $\text{inverse } (\text{star-n } X) = \text{star-n } (\%n. \text{inverse}(X n))$   
 ⟨proof⟩

**lemma** *star-n-le*:

$star-n X \leq star-n Y =$   
 $(\{n. X n \leq Y n\} \in FreeUltrafilterNat)$   
 ⟨proof⟩

**lemma** *star-n-less*:

$star-n X < star-n Y = (\{n. X n < Y n\} \in FreeUltrafilterNat)$   
 ⟨proof⟩

**lemma** *star-n-zero-num*:  $0 = star-n (\%n. 0)$

⟨proof⟩

**lemma** *star-n-one-num*:  $1 = star-n (\%n. 1)$

⟨proof⟩

**lemma** *star-n-abs*:

$abs (star-n X) = star-n (\%n. abs (X n))$   
 ⟨proof⟩

## 7.6 Misc Others

**lemma** *hypreal-not-refl2*:  $!(x::hypreal). x < y ==> x \neq y$

⟨proof⟩

**lemma** *hypreal-eq-minus-iff*:  $((x::hypreal) = y) = (x + - y = 0)$

⟨proof⟩

**lemma** *hypreal-mult-left-cancel*:  $(c::hypreal) \neq 0 ==> (c*a=c*b) = (a=b)$

⟨proof⟩

**lemma** *hypreal-mult-right-cancel*:  $(c::hypreal) \neq 0 ==> (a*c=b*c) = (a=b)$

⟨proof⟩

**lemma** *hypreal-omega-gt-zero [simp]*:  $0 < omega$

⟨proof⟩

## 7.7 Existence of Infinite Hyperreal Number

Existence of infinite number not corresponding to any real number. Use assumption that member  $\mathcal{U}$  is not finite.

A few lemmas first

**lemma** *lemma-omega-empty-singleton-disj*:  $\{n::nat. x = real n\} = \{\} \mid$

$(\exists y. \{n::nat. x = real n\} = \{y\})$

⟨proof⟩

**lemma** *lemma-finite-omega-set*: *finite*  $\{n::nat. x = real n\}$

⟨proof⟩

**lemma** *not-ex-hypreal-of-real-eq-omega*:  
 $\sim (\exists x. \text{hypreal-of-real } x = \text{omega})$   
 ⟨proof⟩

**lemma** *hypreal-of-real-not-eq-omega*: *hypreal-of-real*  $x \neq \text{omega}$   
 ⟨proof⟩

Existence of infinitesimal number also not corresponding to any real number

**lemma** *lemma-epsilon-empty-singleton-disj*:  
 $\{n::\text{nat}. x = \text{inverse}(\text{real}(\text{Suc } n))\} = \{\} \mid$   
 $(\exists y. \{n::\text{nat}. x = \text{inverse}(\text{real}(\text{Suc } n))\} = \{y\})$   
 ⟨proof⟩

**lemma** *lemma-finite-epsilon-set*: *finite*  $\{n. x = \text{inverse}(\text{real}(\text{Suc } n))\}$   
 ⟨proof⟩

**lemma** *not-ex-hypreal-of-real-eq-epsilon*:  $\sim (\exists x. \text{hypreal-of-real } x = \text{epsilon})$   
 ⟨proof⟩

**lemma** *hypreal-of-real-not-eq-epsilon*: *hypreal-of-real*  $x \neq \text{epsilon}$   
 ⟨proof⟩

**lemma** *hypreal-epsilon-not-zero*: *epsilon*  $\neq 0$   
 ⟨proof⟩

**lemma** *hypreal-epsilon-inverse-omega*: *epsilon* = *inverse*(*omega*)  
 ⟨proof⟩

**lemma** *hypreal-epsilon-gt-zero*:  $0 < \text{epsilon}$   
 ⟨proof⟩

## 7.8 Absolute Value Function for the Hyperreals

**lemma** *hrabs-add-less*:  
 $[| \text{abs } x < r; \text{abs } y < s |] \implies \text{abs}(x+y) < r + (s::\text{hypreal})$   
 ⟨proof⟩

**lemma** *hrabs-less-gt-zero*:  $\text{abs } x < r \implies (0::\text{hypreal}) < r$   
 ⟨proof⟩

**lemma** *hrabs-disj*:  $\text{abs } x = (x::'a::\text{abs-if}) \mid \text{abs } x = -x$   
 ⟨proof⟩

**lemma** *hrabs-add-lemma-disj*:  $(y::\text{hypreal}) + -x + (y + -z) = \text{abs } (x + -z)$   
 $\implies y = z \mid x = y$   
 ⟨proof⟩

## 7.9 Embedding the Naturals into the Hyperreals

abbreviation

*hypreal-of-nat* :: nat => hypreal **where**  
*hypreal-of-nat* == of-nat

**lemma** *SNat-eq*: Nats = {n. ∃ N. n = hypreal-of-nat N}  
 ⟨proof⟩

**lemma** *hypreal-of-nat-eq*:  
 hypreal-of-nat (n::nat) = hypreal-of-real (real n)  
 ⟨proof⟩

**lemma** *hypreal-of-nat*:  
 hypreal-of-nat m = star-n (%n. real m)  
 ⟨proof⟩

⟨ML⟩

## 7.10 Exponentials on the Hyperreals

**lemma** *hpowr-0* [simp]:  $r \wedge 0 = (1::hypreal)$   
 ⟨proof⟩

**lemma** *hpowr-Suc* [simp]:  $r \wedge (Suc\ n) = (r::hypreal) * (r \wedge n)$   
 ⟨proof⟩

**lemma** *hrealpow-two*:  $(r::hypreal) \wedge Suc\ (Suc\ 0) = r * r$   
 ⟨proof⟩

**lemma** *hrealpow-two-le* [simp]:  $(0::hypreal) \leq r \wedge Suc\ (Suc\ 0)$   
 ⟨proof⟩

**lemma** *hrealpow-two-le-add-order* [simp]:  
 $(0::hypreal) \leq u \wedge Suc\ (Suc\ 0) + v \wedge Suc\ (Suc\ 0)$   
 ⟨proof⟩

**lemma** *hrealpow-two-le-add-order2* [simp]:  
 $(0::hypreal) \leq u \wedge Suc\ (Suc\ 0) + v \wedge Suc\ (Suc\ 0) + w \wedge Suc\ (Suc\ 0)$   
 ⟨proof⟩

**lemma** *hypreal-add-nonneg-eq-0-iff*:  
 $[| 0 \leq x; 0 \leq y |] ==> (x+y = 0) = (x = 0 \ \& \ y = (0::hypreal))$   
 ⟨proof⟩

FIXME: DELETE THESE

**lemma** *hypreal-three-squares-add-zero-iff*:

$$(x*x + y*y + z*z = 0) = (x = 0 \ \& \ y = 0 \ \& \ z = (0::hypreal))$$

*<proof>*

**lemma** *hrealpow-three-squares-add-zero-iff [simp]*:

$$(x \wedge \text{Suc} (\text{Suc } 0) + y \wedge \text{Suc} (\text{Suc } 0) + z \wedge \text{Suc} (\text{Suc } 0) = (0::hypreal)) =$$

$$(x = 0 \ \& \ y = 0 \ \& \ z = 0)$$

*<proof>*

**lemma** *hrabs-hrealpow-two [simp]*:

$$\text{abs}(x \wedge \text{Suc} (\text{Suc } 0)) = (x::hypreal) \wedge \text{Suc} (\text{Suc } 0)$$

*<proof>*

**lemma** *two-hrealpow-ge-one [simp]*:  $(1::hypreal) \leq 2 \wedge n$

*<proof>*

**lemma** *two-hrealpow-gt [simp]*: *hypreal-of-nat*  $n < 2 \wedge n$

*<proof>*

**lemma** *hrealpow*:

$$\text{star-}n \ X \wedge m = \text{star-}n \ (\%n. (X \ n::\text{real}) \wedge m)$$

*<proof>*

**lemma** *hrealpow-sum-square-expand*:

$$(x + (y::hypreal)) \wedge \text{Suc} (\text{Suc } 0) =$$

$$x \wedge \text{Suc} (\text{Suc } 0) + y \wedge \text{Suc} (\text{Suc } 0) + (\text{hypreal-of-nat} (\text{Suc} (\text{Suc } 0))) * x * y$$

*<proof>*

**lemma** *power-hypreal-of-real-numeral*:

$$(\text{numeral } v \ :: \ \text{hypreal}) \wedge n = \text{hypreal-of-real} ((\text{numeral } v) \wedge n)$$

*<proof>*

**declare** *power-hypreal-of-real-numeral [of - numeral w, simp] for w*

**lemma** *power-hypreal-of-real-neg-numeral*:

$$(\text{neg-numeral } v \ :: \ \text{hypreal}) \wedge n = \text{hypreal-of-real} ((\text{neg-numeral } v) \wedge n)$$

*<proof>*

**declare** *power-hypreal-of-real-neg-numeral [of - numeral w, simp] for w*

## 7.11 Powers with Hypernatural Exponents

**definition** *pow* :: [*'a*::*power star*, *nat star*]  $\Rightarrow$  *'a star* (**infixr** *pow* 80) **where**  
*hyperpow-def [transfer-unfold]:*  $R \ \text{pow } N = ( *f2* \ \text{op } \wedge ) \ R \ N$

**lemma** *Standard-hyperpow [simp]*:

$$[r \in \text{Standard}; n \in \text{Standard}] \Longrightarrow r \ \text{pow } n \in \text{Standard}$$

*<proof>*

**lemma** *hyperpow*:  $\text{star-}n\ X\ \text{pow}\ \text{star-}n\ Y = \text{star-}n\ (\%n.\ X\ n\ \wedge\ Y\ n)$   
 ⟨proof⟩

**lemma** *hyperpow-zero* [simp]:  
 $\bigwedge n. (0::'a::\{\text{power, semiring-0}\}\ \text{star})\ \text{pow}\ (n + (1::\text{hypnat})) = 0$   
 ⟨proof⟩

**lemma** *hyperpow-not-zero*:  
 $\bigwedge r\ n. r \neq (0::'a::\{\text{field}\}\ \text{star}) \implies r\ \text{pow}\ n \neq 0$   
 ⟨proof⟩

**lemma** *hyperpow-inverse*:  
 $\bigwedge r\ n. r \neq (0::'a::\{\text{field-inverse-zero}\}\ \text{star})$   
 $\implies \text{inverse}\ (r\ \text{pow}\ n) = (\text{inverse}\ r)\ \text{pow}\ n$   
 ⟨proof⟩

**lemma** *hyperpow-hrabs*:  
 $\bigwedge r\ n. \text{abs}\ (r::'a::\{\text{linordered-idom}\}\ \text{star})\ \text{pow}\ n = \text{abs}\ (r\ \text{pow}\ n)$   
 ⟨proof⟩

**lemma** *hyperpow-add*:  
 $\bigwedge r\ n\ m. (r::'a::\{\text{monoid-mult}\}\ \text{star})\ \text{pow}\ (n + m) = (r\ \text{pow}\ n) * (r\ \text{pow}\ m)$   
 ⟨proof⟩

**lemma** *hyperpow-one* [simp]:  
 $\bigwedge r. (r::'a::\{\text{monoid-mult}\}\ \text{star})\ \text{pow}\ (1::\text{hypnat}) = r$   
 ⟨proof⟩

**lemma** *hyperpow-two*:  
 $\bigwedge r. (r::'a::\{\text{monoid-mult}\}\ \text{star})\ \text{pow}\ (2::\text{hypnat}) = r * r$   
 ⟨proof⟩

**lemma** *hyperpow-gt-zero*:  
 $\bigwedge r\ n. (0::'a::\{\text{linordered-semidom}\}\ \text{star}) < r \implies 0 < r\ \text{pow}\ n$   
 ⟨proof⟩

**lemma** *hyperpow-ge-zero*:  
 $\bigwedge r\ n. (0::'a::\{\text{linordered-semidom}\}\ \text{star}) \leq r \implies 0 \leq r\ \text{pow}\ n$   
 ⟨proof⟩

**lemma** *hyperpow-le*:  
 $\bigwedge x\ y\ n. \llbracket (0::'a::\{\text{linordered-semidom}\}\ \text{star}) < x; x \leq y \rrbracket$   
 $\implies x\ \text{pow}\ n \leq y\ \text{pow}\ n$   
 ⟨proof⟩

**lemma** *hyperpow-eq-one* [simp]:  
 $\bigwedge n. 1\ \text{pow}\ n = (1::'a::\{\text{monoid-mult}\}\ \text{star})$   
 ⟨proof⟩

**lemma** *hrabs-hyperpow-minus-one* [simp]:  
 $\bigwedge n. \text{abs}(-1 \text{ pow } n) = (1::'a::\{\text{linordered-idom}\} \text{ star})$   
 ⟨proof⟩

**lemma** *hyperpow-mult*:  
 $\bigwedge r \ s \ n. (r * s::'a::\{\text{comm-monoid-mult}\} \text{ star}) \text{ pow } n$   
 $= (r \text{ pow } n) * (s \text{ pow } n)$   
 ⟨proof⟩

**lemma** *hyperpow-two-le* [simp]:  
 $\bigwedge r. (0::'a::\{\text{monoid-mult, linordered-ring-strict}\} \text{ star}) \leq r \text{ pow } 2$   
 ⟨proof⟩

**lemma** *hrabs-hyperpow-two* [simp]:  
 $\text{abs}(x \text{ pow } 2) =$   
 $(x::'a::\{\text{monoid-mult, linordered-ring-strict}\} \text{ star}) \text{ pow } 2$   
 ⟨proof⟩

**lemma** *hyperpow-two-hrabs* [simp]:  
 $\text{abs}(x::'a::\{\text{linordered-idom}\} \text{ star}) \text{ pow } 2 = x \text{ pow } 2$   
 ⟨proof⟩

The precondition could be weakened to  $(0::'a) \leq x$

**lemma** *hypreal-mult-less-mono*:  
 $[[ u < v; \ x < y; \ (0::\text{hypreal}) < v; \ 0 < x ]] \implies u * x < v * y$   
 ⟨proof⟩

**lemma** *hyperpow-two-gt-one*:  
 $\bigwedge r::'a::\{\text{linordered-semidom}\} \text{ star}. 1 < r \implies 1 < r \text{ pow } 2$   
 ⟨proof⟩

**lemma** *hyperpow-two-ge-one*:  
 $\bigwedge r::'a::\{\text{linordered-semidom}\} \text{ star}. 1 \leq r \implies 1 \leq r \text{ pow } 2$   
 ⟨proof⟩

**lemma** *two-hyperpow-ge-one* [simp]:  $(1::\text{hypreal}) \leq 2 \text{ pow } n$   
 ⟨proof⟩

**lemma** *hyperpow-minus-one2* [simp]:  
 $\bigwedge n. -1 \text{ pow } (2 * n) = (1::\text{hypreal})$   
 ⟨proof⟩

**lemma** *hyperpow-less-le*:  
 $!!r \ n \ N. [[(0::\text{hypreal}) \leq r; \ r \leq 1; \ n < N]] \implies r \text{ pow } N \leq r \text{ pow } n$   
 ⟨proof⟩

**lemma** *hyperpow-SHNat-le*:  
 $[[ 0 \leq r; \ r \leq (1::\text{hypreal}); \ N \in \text{HNatInfinite} ]]$   
 $\implies \text{ALL } n: \text{Nats}. r \text{ pow } N \leq r \text{ pow } n$

⟨proof⟩

**lemma** *hyperpow-realpow*:

$$(hypreal-of-real\ r)\ pow\ (hypnat-of-nat\ n) = hypreal-of-real\ (r \wedge n)$$

⟨proof⟩

**lemma** *hyperpow-SReal [simp]*:

$$(hypreal-of-real\ r)\ pow\ (hypnat-of-nat\ n) \in Reals$$

⟨proof⟩

**lemma** *hyperpow-zero-HNatInfinite [simp]*:

$$N \in HNatInfinite \implies (0::hypreal)\ pow\ N = 0$$

⟨proof⟩

**lemma** *hyperpow-le-le*:

$$[(0::hypreal) \leq r; r \leq 1; n \leq N] \implies r\ pow\ N \leq r\ pow\ n$$

⟨proof⟩

**lemma** *hyperpow-Suc-le-self2*:

$$[(0::hypreal) \leq r; r < 1] \implies r\ pow\ (n + (1::hypnat)) \leq r$$

⟨proof⟩

**lemma** *hyperpow-hypnat-of-nat*:  $\bigwedge x. x\ pow\ hypnat-of-nat\ n = x \wedge n$

⟨proof⟩

**lemma** *of-hypreal-hyperpow*:

$$\bigwedge x\ n. of-hypreal\ (x\ pow\ n) = (of-hypreal\ x::'a::\{real-algebra-1\}\ star)\ pow\ n$$

⟨proof⟩

end

## 8 NSA: Infinite Numbers, Infinitesimals, Infinitely Close Relation

**theory** *NSA*

**imports** *HyperDef RComplete*

**begin**

**definition**

$$hnorm :: 'a::real-normed-vector\ star \Rightarrow real\ star\ \mathbf{where}$$

[transfer-unfold]:  $hnorm = *f* norm$

**definition**

$$Infinitesimal :: ('a::real-normed-vector)\ star\ set\ \mathbf{where}$$

$$Infinitesimal = \{x. \forall r \in Reals. 0 < r \longrightarrow hnorm\ x < r\}$$

**definition**

$HFinite :: ('a::real-normed-vector) \text{ star set where}$   
 $HFinite = \{x. \exists r \in Reals. \text{hnorm } x < r\}$

**definition**

$HInfinite :: ('a::real-normed-vector) \text{ star set where}$   
 $HInfinite = \{x. \forall r \in Reals. r < \text{hnorm } x\}$

**definition**

$approx :: ['a::real-normed-vector \text{ star}, 'a \text{ star}] \Rightarrow \text{bool (infixl } @ = 50) \text{ where}$   
 — the ‘infinitely close’ relation  
 $(x @ = y) = ((x - y) \in Infinitesimal)$

**definition**

$st :: hypreal \Rightarrow hypreal \text{ where}$   
 — the standard part of a hyperreal  
 $st = (\%x. @r. x \in HFinite \ \& \ r \in Reals \ \& \ r @ = x)$

**definition**

$monad :: 'a::real-normed-vector \text{ star} \Rightarrow 'a \text{ star set where}$   
 $monad \ x = \{y. x @ = y\}$

**definition**

$galaxy :: 'a::real-normed-vector \text{ star} \Rightarrow 'a \text{ star set where}$   
 $galaxy \ x = \{y. (x + -y) \in HFinite\}$

**notation** (*xsymbols*)

$approx$  (infixl  $\approx$  50)

**notation** (*HTML output*)

$approx$  (infixl  $\approx$  50)

**lemma** *SReal-def*:  $Reals == \{x. \exists r. x = \text{hypreal-of-real } r\}$   
 $\langle \text{proof} \rangle$

## 8.1 Nonstandard Extension of the Norm Function

**definition**

$scaleHR :: \text{real star} \Rightarrow 'a \text{ star} \Rightarrow 'a::real-normed-vector \text{ star where}$   
 $[\text{transfer-unfold}]: \text{scaleHR} = \text{starfun2 } \text{scaleR}$

**lemma** *Standard-hnorm* [simp]:  $x \in \text{Standard} \Longrightarrow \text{hnorm } x \in \text{Standard}$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-norm* [simp]:  $\text{star-of } (\text{norm } x) = \text{hnorm } (\text{star-of } x)$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-ge-zero* [simp]:

$\bigwedge x::'a::real-normed-vector \text{ star. } 0 \leq \text{hnorm } x$

$\langle \text{proof} \rangle$

**lemma** *hnorm-eq-zero* [*simp*]:

$\bigwedge x :: 'a :: \text{real-normed-vector star}. (\text{hnorm } x = 0) = (x = 0)$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-triangle-ineq*:

$\bigwedge x y :: 'a :: \text{real-normed-vector star}. \text{hnorm } (x + y) \leq \text{hnorm } x + \text{hnorm } y$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-triangle-ineq3*:

$\bigwedge x y :: 'a :: \text{real-normed-vector star}. |\text{hnorm } x - \text{hnorm } y| \leq \text{hnorm } (x - y)$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-scaleR*:

$\bigwedge x :: 'a :: \text{real-normed-vector star}.$   
 $\text{hnorm } (a *_{\mathbb{R}} x) = |\text{star-of } a| * \text{hnorm } x$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-scaleHR*:

$\bigwedge a (x :: 'a :: \text{real-normed-vector star}).$   
 $\text{hnorm } (\text{scaleHR } a x) = |a| * \text{hnorm } x$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-mult-ineq*:

$\bigwedge x y :: 'a :: \text{real-normed-algebra star}. \text{hnorm } (x * y) \leq \text{hnorm } x * \text{hnorm } y$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-mult*:

$\bigwedge x y :: 'a :: \text{real-normed-div-algebra star}. \text{hnorm } (x * y) = \text{hnorm } x * \text{hnorm } y$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-hyperpow*:

$\bigwedge (x :: 'a :: \{\text{real-normed-div-algebra}\} \text{ star}) n.$   
 $\text{hnorm } (x \text{ pow } n) = \text{hnorm } x \text{ pow } n$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-one* [*simp*]:

$\text{hnorm } (1 :: 'a :: \text{real-normed-div-algebra star}) = 1$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-zero* [*simp*]:

$\text{hnorm } (0 :: 'a :: \text{real-normed-vector star}) = 0$   
 $\langle \text{proof} \rangle$

**lemma** *zero-less-hnorm-iff* [*simp*]:

$\bigwedge x :: 'a :: \text{real-normed-vector star}. (0 < \text{hnorm } x) = (x \neq 0)$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-minus-cancel* [simp]:

$\bigwedge x::'a::\text{real-normed-vector star}. \text{hnorm } (- x) = \text{hnorm } x$   
 ⟨proof⟩

**lemma** *hnorm-minus-commute*:

$\bigwedge a b::'a::\text{real-normed-vector star}. \text{hnorm } (a - b) = \text{hnorm } (b - a)$   
 ⟨proof⟩

**lemma** *hnorm-triangle-ineq2*:

$\bigwedge a b::'a::\text{real-normed-vector star}. \text{hnorm } a - \text{hnorm } b \leq \text{hnorm } (a - b)$   
 ⟨proof⟩

**lemma** *hnorm-triangle-ineq4*:

$\bigwedge a b::'a::\text{real-normed-vector star}. \text{hnorm } (a - b) \leq \text{hnorm } a + \text{hnorm } b$   
 ⟨proof⟩

**lemma** *abs-hnorm-cancel* [simp]:

$\bigwedge a::'a::\text{real-normed-vector star}. |\text{hnorm } a| = \text{hnorm } a$   
 ⟨proof⟩

**lemma** *hnorm-of-hypreal* [simp]:

$\bigwedge r. \text{hnorm } (\text{of-hypreal } r::'a::\text{real-normed-algebra-1 star}) = |r|$   
 ⟨proof⟩

**lemma** *nonzero-hnorm-inverse*:

$\bigwedge a::'a::\text{real-normed-div-algebra star}.$   
 $a \neq 0 \implies \text{hnorm } (\text{inverse } a) = \text{inverse } (\text{hnorm } a)$   
 ⟨proof⟩

**lemma** *hnorm-inverse*:

$\bigwedge a::'a::\{\text{real-normed-div-algebra, division-ring-inverse-zero}\} \text{ star}.$   
 $\text{hnorm } (\text{inverse } a) = \text{inverse } (\text{hnorm } a)$   
 ⟨proof⟩

**lemma** *hnorm-divide*:

$\bigwedge a b::'a::\{\text{real-normed-field, field-inverse-zero}\} \text{ star}.$   
 $\text{hnorm } (a / b) = \text{hnorm } a / \text{hnorm } b$   
 ⟨proof⟩

**lemma** *hypreal-hnorm-def* [simp]:

$\bigwedge r::\text{hypreal}. \text{hnorm } r = |r|$   
 ⟨proof⟩

**lemma** *hnorm-add-less*:

$\bigwedge (x::'a::\text{real-normed-vector star}) y r s.$   
 $\llbracket \text{hnorm } x < r; \text{hnorm } y < s \rrbracket \implies \text{hnorm } (x + y) < r + s$   
 ⟨proof⟩

**lemma** *hnorm-mult-less*:

$\bigwedge (x::'a::\text{real-normed-algebra star}) y r s.$   
 $\llbracket \text{hnorm } x < r; \text{hnorm } y < s \rrbracket \implies \text{hnorm } (x * y) < r * s$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-scaleHR-less*:  
 $\llbracket |x| < r; \text{hnorm } y < s \rrbracket \implies \text{hnorm } (\text{scaleHR } x y) < r * s$   
 $\langle \text{proof} \rangle$

## 8.2 Closure Laws for the Standard Reals

**lemma** *Reals-minus-iff* [simp]:  $(-x \in \text{Reals}) = (x \in \text{Reals})$   
 $\langle \text{proof} \rangle$

**lemma** *Reals-add-cancel*:  $\llbracket x + y \in \text{Reals}; y \in \text{Reals} \rrbracket \implies x \in \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-hrabs*:  $(x::\text{hypreal}) \in \text{Reals} \implies \text{abs } x \in \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-hypreal-of-real* [simp]:  $\text{hypreal-of-real } x \in \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-divide-numeral*:  $r \in \text{Reals} \implies r / (\text{numeral } w::\text{hypreal}) \in \text{Reals}$   
 $\langle \text{proof} \rangle$

epsilon is not in Reals because it is an infinitesimal

**lemma** *SReal-epsilon-not-mem*:  $\text{epsilon} \notin \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-omega-not-mem*:  $\text{omega} \notin \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-UNIV-real*:  $\{x. \text{hypreal-of-real } x \in \text{Reals}\} = (\text{UNIV}::\text{real set})$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-iff*:  $(x \in \text{Reals}) = (\exists y. x = \text{hypreal-of-real } y)$   
 $\langle \text{proof} \rangle$

**lemma** *hypreal-of-real-image*:  $\text{hypreal-of-real } ` (\text{UNIV}::\text{real set}) = \text{Reals}$   
 $\langle \text{proof} \rangle$

**lemma** *inv-hypreal-of-real-image*:  $\text{inv } \text{hypreal-of-real } ` \text{Reals} = \text{UNIV}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-hypreal-of-real-image*:  
 $\llbracket \exists x. x: P; P \subseteq \text{Reals} \rrbracket \implies \exists Q. P = \text{hypreal-of-real } ` Q$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-dense*:

$\llbracket (x::\text{hypreal}) \in \text{Reals}; y \in \text{Reals}; x < y \rrbracket \implies \exists r \in \text{Reals}. x < r \ \& \ r < y$   
 <proof>

Completeness of Reals, but both lemmas are unused.

**lemma** *SReal-sup-lemma*:

$P \subseteq \text{Reals} \implies ((\exists x \in P. y < x) =$   
 $(\exists X. \text{hypreal-of-real } X \in P \ \& \ y < \text{hypreal-of-real } X))$   
 <proof>

**lemma** *SReal-sup-lemma2*:

$\llbracket P \subseteq \text{Reals}; \exists x. x \in P; \exists y \in \text{Reals}. \forall x \in P. x < y \rrbracket$   
 $\implies (\exists X. X \in \{w. \text{hypreal-of-real } w \in P\}) \ \&$   
 $(\exists Y. \forall X \in \{w. \text{hypreal-of-real } w \in P\}. X < Y)$   
 <proof>

### 8.3 Set of Finite Elements is a Subring of the Extended Reals

**lemma** *HFinite-add*:  $\llbracket x \in \text{HFinite}; y \in \text{HFinite} \rrbracket \implies (x+y) \in \text{HFinite}$   
 <proof>

**lemma** *HFinite-mult*:

**fixes**  $x \ y :: 'a::\text{real-normed-algebra } \text{star}$   
**shows**  $\llbracket x \in \text{HFinite}; y \in \text{HFinite} \rrbracket \implies x * y \in \text{HFinite}$   
 <proof>

**lemma** *HFinite-scaleHR*:

$\llbracket x \in \text{HFinite}; y \in \text{HFinite} \rrbracket \implies \text{scaleHR } x \ y \in \text{HFinite}$   
 <proof>

**lemma** *HFinite-minus-iff*:  $(-x \in \text{HFinite}) = (x \in \text{HFinite})$   
 <proof>

**lemma** *HFinite-star-of [simp]*:  $\text{star-of } x \in \text{HFinite}$   
 <proof>

**lemma** *SReal-subset-HFinite*:  $(\text{Reals}::\text{hypreal set}) \subseteq \text{HFinite}$   
 <proof>

**lemma** *HFiniteD*:  $x \in \text{HFinite} \implies \exists t \in \text{Reals}. \text{hnorm } x < t$   
 <proof>

**lemma** *HFinite-hrabs-iff [iff]*:  $(\text{abs } (x::\text{hypreal}) \in \text{HFinite}) = (x \in \text{HFinite})$   
 <proof>

**lemma** *HFinite-hnorm-iff [iff]*:

$(\text{hnorm } (x::\text{hypreal}) \in \text{HFinite}) = (x \in \text{HFinite})$   
 <proof>

**lemma** *HFinite-numeral [simp]*:  $\text{numeral } w \in \text{HFinite}$

*<proof>*

**lemma** *HFinite-0* [simp]:  $0 \in \text{HFinite}$   
*<proof>*

**lemma** *HFinite-1* [simp]:  $1 \in \text{HFinite}$   
*<proof>*

**lemma** *hrealpow-HFinite*:  
**fixes**  $x :: 'a :: \{\text{real-normed-algebra}, \text{monoid-mult}\}$  *star*  
**shows**  $x \in \text{HFinite} \implies x \wedge n \in \text{HFinite}$   
*<proof>*

**lemma** *HFinite-bounded*:  
 $\llbracket (x :: \text{hypreal}) \in \text{HFinite}; y \leq x; 0 \leq y \rrbracket \implies y \in \text{HFinite}$   
*<proof>*

## 8.4 Set of Infinitesimals is a Subring of the Hyperreals

**lemma** *InfinitesimalI*:  
 $(\bigwedge r. \llbracket r \in \mathbb{R}; 0 < r \rrbracket \implies \text{hnorm } x < r) \implies x \in \text{Infinitesimal}$   
*<proof>*

**lemma** *InfinitesimalD*:  
 $x \in \text{Infinitesimal} \implies \forall r \in \text{Reals}. 0 < r \dashrightarrow \text{hnorm } x < r$   
*<proof>*

**lemma** *InfinitesimalI2*:  
 $(\bigwedge r. 0 < r \implies \text{hnorm } x < \text{star-of } r) \implies x \in \text{Infinitesimal}$   
*<proof>*

**lemma** *InfinitesimalD2*:  
 $\llbracket x \in \text{Infinitesimal}; 0 < r \rrbracket \implies \text{hnorm } x < \text{star-of } r$   
*<proof>*

**lemma** *Infinitesimal-zero* [iff]:  $0 \in \text{Infinitesimal}$   
*<proof>*

**lemma** *hypreal-sum-of-halves*:  $x / (2 :: \text{hypreal}) + x / (2 :: \text{hypreal}) = x$   
*<proof>*

**lemma** *Infinitesimal-add*:  
 $\llbracket x \in \text{Infinitesimal}; y \in \text{Infinitesimal} \rrbracket \implies (x+y) \in \text{Infinitesimal}$   
*<proof>*

**lemma** *Infinitesimal-minus-iff* [simp]:  $(-x : \text{Infinitesimal}) = (x : \text{Infinitesimal})$   
*<proof>*

**lemma** *Infinitesimal-hnorm-iff*:

$(\text{hnorm } x \in \text{Infinitesimal}) = (x \in \text{Infinitesimal})$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-hrabs-iff [iff]*:

$(\text{abs } (x::\text{hypreal}) \in \text{Infinitesimal}) = (x \in \text{Infinitesimal})$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-of-hypreal-iff [simp]*:

$((\text{of-hypreal } x::'a::\text{real-normed-algebra-1 star}) \in \text{Infinitesimal}) =$   
 $(x \in \text{Infinitesimal})$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-diff*:

$[[ x \in \text{Infinitesimal}; y \in \text{Infinitesimal} ]] ==> x - y \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-mult*:

**fixes**  $x y :: 'a::\text{real-normed-algebra star}$   
**shows**  $[[ x \in \text{Infinitesimal}; y \in \text{Infinitesimal} ]] ==> (x * y) \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-HFinite-mult*:

**fixes**  $x y :: 'a::\text{real-normed-algebra star}$   
**shows**  $[[ x \in \text{Infinitesimal}; y \in \text{HFinite} ]] ==> (x * y) \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-HFinite-scaleHR*:

$[[ x \in \text{Infinitesimal}; y \in \text{HFinite} ]] ==> \text{scaleHR } x y \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-HFinite-mult2*:

**fixes**  $x y :: 'a::\text{real-normed-algebra star}$   
**shows**  $[[ x \in \text{Infinitesimal}; y \in \text{HFinite} ]] ==> (y * x) \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-scaleR2*:

$x \in \text{Infinitesimal} ==> a *_{\mathbb{R}} x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Compl-HFinite: - HFinite = HInfinite*

$\langle \text{proof} \rangle$

**lemma** *HInfinite-inverse-Infinitesimal*:

**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $x \in \text{HInfinite} ==> \text{inverse } x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *HInfiniteI*:  $(\bigwedge r. r \in \mathbb{R} \implies r < \text{hnorm } x) \implies x \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfiniteD*:  $\llbracket x \in \text{HInfinite}; r \in \mathbb{R} \rrbracket \implies r < \text{hnorm } x$   
 ⟨proof⟩

**lemma** *HInfinite-mult*:  
**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $\llbracket x \in \text{HInfinite}; y \in \text{HInfinite} \rrbracket \implies (x*y) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *hypreal-add-zero-less-le-mono*:  $\llbracket r < x; (0::\text{hypreal}) \leq y \rrbracket \implies r < x+y$   
 ⟨proof⟩

**lemma** *HInfinite-add-ge-zero*:  
 $\llbracket (x::\text{hypreal}) \in \text{HInfinite}; 0 \leq y; 0 \leq x \rrbracket \implies (x + y) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfinite-add-ge-zero2*:  
 $\llbracket (x::\text{hypreal}) \in \text{HInfinite}; 0 \leq y; 0 \leq x \rrbracket \implies (y + x) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfinite-add-gt-zero*:  
 $\llbracket (x::\text{hypreal}) \in \text{HInfinite}; 0 < y; 0 < x \rrbracket \implies (x + y) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfinite-minus-iff*:  $(-x \in \text{HInfinite}) = (x \in \text{HInfinite})$   
 ⟨proof⟩

**lemma** *HInfinite-add-le-zero*:  
 $\llbracket (x::\text{hypreal}) \in \text{HInfinite}; y \leq 0; x \leq 0 \rrbracket \implies (x + y) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfinite-add-lt-zero*:  
 $\llbracket (x::\text{hypreal}) \in \text{HInfinite}; y < 0; x < 0 \rrbracket \implies (x + y) \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HFinite-sum-squares*:  
**fixes**  $a b c :: 'a::\text{real-normed-algebra star}$   
**shows**  $\llbracket a: \text{HFinite}; b: \text{HFinite}; c: \text{HFinite} \rrbracket$   
 $\implies a*a + b*b + c*c \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *not-Infinitesimal-not-zero*:  $x \notin \text{Infinitesimal} \implies x \neq 0$   
 ⟨proof⟩

**lemma** *not-Infinitesimal-not-zero2*:  $x \in \text{HFinite} - \text{Infinitesimal} \implies x \neq 0$   
 ⟨proof⟩

**lemma** *HFinite-diff-Infinitesimal-hrabs*:

$(x::\text{hypreal}) \in \text{HFinite} - \text{Infinitesimal} \implies \text{abs } x \in \text{HFinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-le-Infinitesimal*:

$\llbracket e \in \text{Infinitesimal}; \text{hnorm } x \leq e \rrbracket \implies x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hnorm-less-Infinitesimal*:

$\llbracket e \in \text{Infinitesimal}; \text{hnorm } x < e \rrbracket \implies x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hrabs-le-Infinitesimal*:

$\llbracket e \in \text{Infinitesimal}; \text{abs } (x::\text{hypreal}) \leq e \rrbracket \implies x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hrabs-less-Infinitesimal*:

$\llbracket e \in \text{Infinitesimal}; \text{abs } (x::\text{hypreal}) < e \rrbracket \implies x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-interval*:

$\llbracket e \in \text{Infinitesimal}; e' \in \text{Infinitesimal}; e' < x ; x < e \rrbracket$   
 $\implies (x::\text{hypreal}) \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-interval2*:

$\llbracket e \in \text{Infinitesimal}; e' \in \text{Infinitesimal};$   
 $e' \leq x ; x \leq e \rrbracket \implies (x::\text{hypreal}) \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-Infinitesimal-hyperpow*:

$\llbracket (x::\text{hypreal}) \in \text{Infinitesimal}; 0 < N \rrbracket \implies \text{abs } (x \text{ pow } N) \leq \text{abs } x$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-hyperpow*:

$\llbracket (x::\text{hypreal}) \in \text{Infinitesimal}; 0 < N \rrbracket \implies x \text{ pow } N \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hrealpow-hyperpow-Infinitesimal-iff*:

$(x \wedge n \in \text{Infinitesimal}) = (x \text{ pow } (\text{hypnat-of-nat } n) \in \text{Infinitesimal})$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-hrealpow*:

$\llbracket (x::\text{hypreal}) \in \text{Infinitesimal}; 0 < n \rrbracket \implies x \wedge n \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *not-Infinitesimal-mult*:

**fixes**  $x \ y :: 'a::\text{real-normed-div-algebra } \text{star}$

**shows**  $[| x \notin \text{Infinitesimal}; y \notin \text{Infinitesimal} |] \implies (x*y) \notin \text{Infinitesimal}$   
 <proof>

**lemma** *Infinitesimal-mult-disj*:

**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$

**shows**  $x*y \in \text{Infinitesimal} \implies x \in \text{Infinitesimal} \mid y \in \text{Infinitesimal}$

<proof>

**lemma** *HFinite-Infinitesimal-not-zero*:  $x \in \text{HFinite} - \text{Infinitesimal} \implies x \neq 0$

<proof>

**lemma** *HFinite-Infinitesimal-diff-mult*:

**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$

**shows**  $[| x \in \text{HFinite} - \text{Infinitesimal};$

$y \in \text{HFinite} - \text{Infinitesimal}$

$|] \implies (x*y) \in \text{HFinite} - \text{Infinitesimal}$

<proof>

**lemma** *Infinitesimal-subset-HFinite*:

$\text{Infinitesimal} \subseteq \text{HFinite}$

<proof>

**lemma** *Infinitesimal-star-of-mult*:

**fixes**  $x :: 'a::\text{real-normed-algebra star}$

**shows**  $x \in \text{Infinitesimal} \implies x * \text{star-of } r \in \text{Infinitesimal}$

<proof>

**lemma** *Infinitesimal-star-of-mult2*:

**fixes**  $x :: 'a::\text{real-normed-algebra star}$

**shows**  $x \in \text{Infinitesimal} \implies \text{star-of } r * x \in \text{Infinitesimal}$

<proof>

## 8.5 The Infinitely Close Relation

**lemma** *mem-infmal-iff*:  $(x \in \text{Infinitesimal}) = (x @= 0)$

<proof>

**lemma** *approx-minus-iff*:  $(x @= y) = (x - y @= 0)$

<proof>

**lemma** *approx-minus-iff2*:  $(x @= y) = (-y + x @= 0)$

<proof>

**lemma** *approx-refl [iff]*:  $x @= x$

<proof>

**lemma** *hypreal-minus-distrib1*:  $-(y + -(x::'a::\text{ab-group-add})) = x + -y$

<proof>

**lemma** *approx-sym*:  $x \text{ @=} y \implies y \text{ @=} x$   
 ⟨proof⟩

**lemma** *approx-trans*:  $[x \text{ @=} y; y \text{ @=} z] \implies x \text{ @=} z$   
 ⟨proof⟩

**lemma** *approx-trans2*:  $[r \text{ @=} x; s \text{ @=} x] \implies r \text{ @=} s$   
 ⟨proof⟩

**lemma** *approx-trans3*:  $[x \text{ @=} r; x \text{ @=} s] \implies r \text{ @=} s$   
 ⟨proof⟩

**lemma** *approx-reorient*:  $(x \text{ @=} y) = (y \text{ @=} x)$   
 ⟨proof⟩

⟨ML⟩

**lemma** *Infinitesimal-approx-minus*:  $(x - y \in \text{Infinitesimal}) = (x \text{ @=} y)$   
 ⟨proof⟩

**lemma** *approx-monad-iff*:  $(x \text{ @=} y) = (\text{monad}(x) = \text{monad}(y))$   
 ⟨proof⟩

**lemma** *Infinitesimal-approx*:  
 $[x \in \text{Infinitesimal}; y \in \text{Infinitesimal}] \implies x \text{ @=} y$   
 ⟨proof⟩

**lemma** *approx-add*:  $[a \text{ @=} b; c \text{ @=} d] \implies a + c \text{ @=} b + d$   
 ⟨proof⟩

**lemma** *approx-minus*:  $a \text{ @=} b \implies -a \text{ @=} -b$   
 ⟨proof⟩

**lemma** *approx-minus2*:  $-a \text{ @=} -b \implies a \text{ @=} b$   
 ⟨proof⟩

**lemma** *approx-minus-cancel* [*simp*]:  $(-a \text{ @=} -b) = (a \text{ @=} b)$   
 ⟨proof⟩

**lemma** *approx-add-minus*:  $[a \text{ @=} b; c \text{ @=} d] \implies a + -c \text{ @=} b + -d$   
 ⟨proof⟩

**lemma** *approx-diff*:  $[a \text{ @=} b; c \text{ @=} d] \implies a - c \text{ @=} b - d$   
 ⟨proof⟩

**lemma** *approx-mult1*:  
 fixes  $a b c :: 'a::\text{real-normed-algebra star}$   
 shows  $[a \text{ @=} b; c: \text{HFinite}] \implies a * c \text{ @=} b * c$

*<proof>*

**lemma** *approx-mult2*:

**fixes**  $a\ b\ c :: 'a::\text{real-normed-algebra star}$

**shows**  $[| a\ @ = b; c \in \text{HFinite} |] ==> c*a\ @ = c*b$

*<proof>*

**lemma** *approx-mult-subst*:

**fixes**  $u\ v\ x\ y :: 'a::\text{real-normed-algebra star}$

**shows**  $[| u\ @ = v*x; x\ @ = y; v \in \text{HFinite} |] ==> u\ @ = v*y$

*<proof>*

**lemma** *approx-mult-subst2*:

**fixes**  $u\ v\ x\ y :: 'a::\text{real-normed-algebra star}$

**shows**  $[| u\ @ = x*v; x\ @ = y; v \in \text{HFinite} |] ==> u\ @ = y*v$

*<proof>*

**lemma** *approx-mult-subst-star-of*:

**fixes**  $u\ x\ y :: 'a::\text{real-normed-algebra star}$

**shows**  $[| u\ @ = x*\text{star-of } v; x\ @ = y |] ==> u\ @ = y*\text{star-of } v$

*<proof>*

**lemma** *approx-eq-imp*:  $a = b ==> a\ @ = b$

*<proof>*

**lemma** *Infinitesimal-minus-approx*:  $x \in \text{Infinitesimal} ==> -x\ @ = x$

*<proof>*

**lemma** *bex-Infinitesimal-iff*:  $(\exists y \in \text{Infinitesimal}. x - z = y) = (x\ @ = z)$

*<proof>*

**lemma** *bex-Infinitesimal-iff2*:  $(\exists y \in \text{Infinitesimal}. x = z + y) = (x\ @ = z)$

*<proof>*

**lemma** *Infinitesimal-add-approx*:  $[| y \in \text{Infinitesimal}; x + y = z |] ==> x\ @ = z$

*<proof>*

**lemma** *Infinitesimal-add-approx-self*:  $y \in \text{Infinitesimal} ==> x\ @ = x + y$

*<proof>*

**lemma** *Infinitesimal-add-approx-self2*:  $y \in \text{Infinitesimal} ==> x\ @ = y + x$

*<proof>*

**lemma** *Infinitesimal-add-minus-approx-self*:  $y \in \text{Infinitesimal} ==> x\ @ = x - y$

*<proof>*

**lemma** *Infinitesimal-add-cancel*:  $[| y \in \text{Infinitesimal}; x+y\ @ = z |] ==> x\ @ = z$

*<proof>*

**lemma** *Infinitesimal-add-right-cancel:*

$\llbracket y \in \text{Infinitesimal}; x \text{ @} = z + y \rrbracket \implies x \text{ @} = z$   
 $\langle \text{proof} \rangle$

**lemma** *approx-add-left-cancel:*  $d + b \text{ @} = d + c \implies b \text{ @} = c$

$\langle \text{proof} \rangle$

**lemma** *approx-add-right-cancel:*  $b + d \text{ @} = c + d \implies b \text{ @} = c$

$\langle \text{proof} \rangle$

**lemma** *approx-add-mono1:*  $b \text{ @} = c \implies d + b \text{ @} = d + c$

$\langle \text{proof} \rangle$

**lemma** *approx-add-mono2:*  $b \text{ @} = c \implies b + a \text{ @} = c + a$

$\langle \text{proof} \rangle$

**lemma** *approx-add-left-iff [simp]:*  $(a + b \text{ @} = a + c) = (b \text{ @} = c)$

$\langle \text{proof} \rangle$

**lemma** *approx-add-right-iff [simp]:*  $(b + a \text{ @} = c + a) = (b \text{ @} = c)$

$\langle \text{proof} \rangle$

**lemma** *approx-HFinite:*  $\llbracket x \in \text{HFinite}; x \text{ @} = y \rrbracket \implies y \in \text{HFinite}$

$\langle \text{proof} \rangle$

**lemma** *approx-star-of-HFinite:*  $x \text{ @} = \text{star-of } D \implies x \in \text{HFinite}$

$\langle \text{proof} \rangle$

**lemma** *approx-mult-HFinite:*

**fixes**  $a \ b \ c \ d :: 'a::\text{real-normed-algebra star}$

**shows**  $\llbracket a \text{ @} = b; c \text{ @} = d; b \in \text{HFinite}; d \in \text{HFinite} \rrbracket \implies a * c \text{ @} = b * d$   
 $\langle \text{proof} \rangle$

**lemma** *scaleHR-left-diff-distrib:*

$\bigwedge a \ b \ x. \text{scaleHR } (a - b) \ x = \text{scaleHR } a \ x - \text{scaleHR } b \ x$   
 $\langle \text{proof} \rangle$

**lemma** *approx-scaleR1:*

$\llbracket a \text{ @} = \text{star-of } b; c \in \text{HFinite} \rrbracket \implies \text{scaleHR } a \ c \text{ @} = b *_R c$   
 $\langle \text{proof} \rangle$

**lemma** *approx-scaleR2:*

$a \text{ @} = b \implies c *_R a \text{ @} = c *_R b$   
 $\langle \text{proof} \rangle$

**lemma** *approx-scaleR-HFinite:*

$\llbracket a \text{ @} = \text{star-of } b; c \text{ @} = d; d \in \text{HFinite} \rrbracket \implies \text{scaleHR } a \ c \text{ @} = b *_R d$   
 $\langle \text{proof} \rangle$

**lemma** *approx-mult-star-of*:

**fixes**  $a\ c :: 'a::\text{real-normed-algebra}\ \text{star}$

**shows**  $[[a\ @= \text{star-of}\ b; c\ @= \text{star-of}\ d\ ]]$

$==> a*c\ @= \text{star-of}\ b*\text{star-of}\ d$

$\langle\text{proof}\rangle$

**lemma** *approx-SReal-mult-cancel-zero*:

$[[ (a::\text{hypreal}) \in \text{Reals}; a \neq 0; a*x\ @= 0\ ]]$   $==> x\ @= 0$

$\langle\text{proof}\rangle$

**lemma** *approx-mult-SReal1*:  $[[ (a::\text{hypreal}) \in \text{Reals}; x\ @= 0\ ]]$   $==> x*a\ @= 0$

$\langle\text{proof}\rangle$

**lemma** *approx-mult-SReal2*:  $[[ (a::\text{hypreal}) \in \text{Reals}; x\ @= 0\ ]]$   $==> a*x\ @= 0$

$\langle\text{proof}\rangle$

**lemma** *approx-mult-SReal-zero-cancel-iff* [simp]:

$[[ (a::\text{hypreal}) \in \text{Reals}; a \neq 0\ ]]$   $==> (a*x\ @= 0) = (x\ @= 0)$

$\langle\text{proof}\rangle$

**lemma** *approx-SReal-mult-cancel*:

$[[ (a::\text{hypreal}) \in \text{Reals}; a \neq 0; a* w\ @= a*z\ ]]$   $==> w\ @= z$

$\langle\text{proof}\rangle$

**lemma** *approx-SReal-mult-cancel-iff1* [simp]:

$[[ (a::\text{hypreal}) \in \text{Reals}; a \neq 0\ ]]$   $==> (a* w\ @= a*z) = (w\ @= z)$

$\langle\text{proof}\rangle$

**lemma** *approx-le-bound*:  $[[ (z::\text{hypreal}) \leq f; f\ @= g; g \leq z\ ]]$   $==> f\ @= z$

$\langle\text{proof}\rangle$

**lemma** *approx-hnorm*:

**fixes**  $x\ y :: 'a::\text{real-normed-vector}\ \text{star}$

**shows**  $x \approx y \implies \text{hnorm}\ x \approx \text{hnorm}\ y$

$\langle\text{proof}\rangle$

## 8.6 Zero is the Only Infinitesimal that is also a Real

**lemma** *Infinitesimal-less-SReal*:

$[[ (x::\text{hypreal}) \in \text{Reals}; y \in \text{Infinitesimal}; 0 < x\ ]]$   $==> y < x$

$\langle\text{proof}\rangle$

**lemma** *Infinitesimal-less-SReal2*:

$(y::\text{hypreal}) \in \text{Infinitesimal} ==> \forall r \in \text{Reals}. 0 < r \implies y < r$

$\langle\text{proof}\rangle$

**lemma** *SReal-not-Infinitesimal*:

$[[ 0 < y; (y::\text{hypreal}) \in \text{Reals}\ ]]$   $==> y \notin \text{Infinitesimal}$

$\langle\text{proof}\rangle$

**lemma** *SReal-minus-not-Infinitesimal*:

$\llbracket y < 0; (y::\text{hypreal}) \in \text{Reals} \rrbracket \implies y \notin \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-Int-Infinitesimal-zero*:  $\text{Reals Int Infinitesimal} = \{0::\text{hypreal}\}$

$\langle \text{proof} \rangle$

**lemma** *SReal-Infinitesimal-zero*:

$\llbracket (x::\text{hypreal}) \in \text{Reals}; x \in \text{Infinitesimal} \rrbracket \implies x = 0$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-HFfinite-diff-Infinitesimal*:

$\llbracket (x::\text{hypreal}) \in \text{Reals}; x \neq 0 \rrbracket \implies x \in \text{HFfinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *hypreal-of-real-HFfinite-diff-Infinitesimal*:

$\text{hypreal-of-real } x \neq 0 \implies \text{hypreal-of-real } x \in \text{HFfinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-Infinitesimal-iff-0* [iff]:

$(\text{star-of } x \in \text{Infinitesimal}) = (x = 0)$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-HFfinite-diff-Infinitesimal*:

$x \neq 0 \implies \text{star-of } x \in \text{HFfinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *numeral-not-Infinitesimal* [simp]:

$\text{numeral } w \neq (0::\text{hypreal}) \implies (\text{numeral } w :: \text{hypreal}) \notin \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *one-not-Infinitesimal* [simp]:

$(1::'a::\{\text{real-normed-vector}, \text{zero-neq-one}\} \text{star}) \notin \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *approx-SReal-not-zero*:

$\llbracket (y::\text{hypreal}) \in \text{Reals}; x @= y; y \neq 0 \rrbracket \implies x \neq 0$   
 $\langle \text{proof} \rangle$

**lemma** *HFfinite-diff-Infinitesimal-approx*:

$\llbracket x @= y; y \in \text{HFfinite} - \text{Infinitesimal} \rrbracket$   
 $\implies x \in \text{HFfinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-ratio*:

**fixes**  $x y :: 'a::\{\text{real-normed-div-algebra}, \text{field}\} \text{star}$

**shows**  $[| y \neq 0; y \in \text{Infinitesimal}; x/y \in \text{HFinite} |]$   
 $\implies x \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-SReal-divide*:

$[| (x::\text{hypreal}) \in \text{Infinitesimal}; y \in \text{Reals} |] \implies x/y \in \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

## 8.7 Uniqueness: Two Infinitely Close Reals are Equal

**lemma** *star-of-approx-iff* [simp]:  $(\text{star-of } x \text{ @=} \text{star-of } y) = (x = y)$   
 $\langle \text{proof} \rangle$

**lemma** *SReal-approx-iff*:

$[| (x::\text{hypreal}) \in \text{Reals}; y \in \text{Reals} |] \implies (x \text{ @=} y) = (x = y)$   
 $\langle \text{proof} \rangle$

**lemma** *numeral-approx-iff* [simp]:

$(\text{numeral } v \text{ @=} (\text{numeral } w :: 'a::\{\text{numeral,real-normed-vector}\} \text{star})) =$   
 $(\text{numeral } v = (\text{numeral } w :: 'a))$   
 $\langle \text{proof} \rangle$

**lemma** [simp]:

$(\text{numeral } w \text{ @=} (0::'a::\{\text{numeral,real-normed-vector}\} \text{star})) =$   
 $(\text{numeral } w = (0::'a))$   
 $((0::'a::\{\text{numeral,real-normed-vector}\} \text{star}) \text{ @=} \text{numeral } w) =$   
 $(\text{numeral } w = (0::'a))$   
 $(\text{numeral } w \text{ @=} (1::'b::\{\text{numeral,one,real-normed-vector}\} \text{star})) =$   
 $(\text{numeral } w = (1::'b))$   
 $((1::'b::\{\text{numeral,one,real-normed-vector}\} \text{star}) \text{ @=} \text{numeral } w) =$   
 $(\text{numeral } w = (1::'b))$   
 $\sim (0 \text{ @=} (1::'c::\{\text{zero-neq-one,real-normed-vector}\} \text{star}))$   
 $\sim (1 \text{ @=} (0::'c::\{\text{zero-neq-one,real-normed-vector}\} \text{star}))$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-approx-numeral-iff* [simp]:

$(\text{star-of } k \text{ @=} \text{numeral } w) = (k = \text{numeral } w)$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-approx-zero-iff* [simp]:  $(\text{star-of } k \text{ @=} 0) = (k = 0)$

$\langle \text{proof} \rangle$

**lemma** *star-of-approx-one-iff* [simp]:  $(\text{star-of } k \text{ @=} 1) = (k = 1)$

$\langle \text{proof} \rangle$

**lemma** *approx-unique-real*:

$[| (r::\text{hypreal}) \in \text{Reals}; s \in \text{Reals}; r \text{ @=} x; s \text{ @=} x |] \implies r = s$   
 $\langle \text{proof} \rangle$

## 8.8 Existence of Unique Real Infinitely Close

### 8.8.1 Lifting of the Ub and Lub Properties

**lemma** *hypreal-of-real-isUb-iff*:

$$(isUb (Reals) (hypreal-of-real ' Q) (hypreal-of-real Y)) = (isUb (UNIV :: real set) Q Y)$$

*<proof>*

**lemma** *hypreal-of-real-isLub1*:

$$isLub Reals (hypreal-of-real ' Q) (hypreal-of-real Y) ==> isLub (UNIV :: real set) Q Y$$

*<proof>*

**lemma** *hypreal-of-real-isLub2*:

$$isLub (UNIV :: real set) Q Y ==> isLub Reals (hypreal-of-real ' Q) (hypreal-of-real Y)$$

*<proof>*

**lemma** *hypreal-of-real-isLub-iff*:

$$(isLub Reals (hypreal-of-real ' Q) (hypreal-of-real Y)) = (isLub (UNIV :: real set) Q Y)$$

*<proof>*

**lemma** *lemma-isUb-hypreal-of-real*:

$$isUb Reals P Y ==> \exists Yo. isUb Reals P (hypreal-of-real Yo)$$

*<proof>*

**lemma** *lemma-isLub-hypreal-of-real*:

$$isLub Reals P Y ==> \exists Yo. isLub Reals P (hypreal-of-real Yo)$$

*<proof>*

**lemma** *lemma-isLub-hypreal-of-real2*:

$$\exists Yo. isLub Reals P (hypreal-of-real Yo) ==> \exists Y. isLub Reals P Y$$

*<proof>*

**lemma** *SReal-complete*:

$$[| P \subseteq Reals; \exists x. x \in P; \exists Y. isUb Reals P Y |] ==> \exists t::hypreal. isLub Reals P t$$

*<proof>*

**lemma** *hypreal-isLub-unique*:

$$[| isLub R S x; isLub R S y |] ==> x = (y::hypreal)$$

*<proof>*

**lemma** *lemma-st-part-ub*:

$$(x::hypreal) \in HFinite ==> \exists u. isUb Reals \{s. s \in Reals \ \& \ s < x\} u$$

*<proof>*

**lemma** *lemma-st-part-nonempty*:

$(x::\text{hypreal}) \in \text{HFinite} \implies \exists y. y \in \{s. s \in \text{Reals} \ \& \ s < x\}$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-st-part-subset*:  $\{s. s \in \text{Reals} \ \& \ s < x\} \subseteq \text{Reals}$

$\langle \text{proof} \rangle$

**lemma** *lemma-st-part-lub*:

$(x::\text{hypreal}) \in \text{HFinite} \implies \exists t. \text{isLub } \text{Reals} \ \{s. s \in \text{Reals} \ \& \ s < x\} \ t$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-hypreal-le-left-cancel*:  $((t::\text{hypreal}) + r \leq t) = (r \leq 0)$

$\langle \text{proof} \rangle$

**lemma** *lemma-st-part-le1*:

$[\![ (x::\text{hypreal}) \in \text{HFinite}; \text{isLub } \text{Reals} \ \{s. s \in \text{Reals} \ \& \ s < x\} \ t;$   
 $r \in \text{Reals}; \ 0 < r \ ]\!] \implies x \leq t + r$   
 $\langle \text{proof} \rangle$

**lemma** *hypreal-settle-less-trans*:

$[\![ S * \leq (x::\text{hypreal}); x < y \ ]\!] \implies S * \leq y$   
 $\langle \text{proof} \rangle$

**lemma** *hypreal-gt-isUb*:

$[\![ \text{isUb } R \ S \ (x::\text{hypreal}); x < y; y \in R \ ]\!] \implies \text{isUb } R \ S \ y$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-st-part-gt-ub*:

$[\![ (x::\text{hypreal}) \in \text{HFinite}; x < y; y \in \text{Reals} \ ]\!] \implies \text{isUb } \text{Reals} \ \{s. s \in \text{Reals} \ \& \ s < x\} \ y$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-minus-le-zero*:  $t \leq t + -r \implies r \leq (0::\text{hypreal})$

$\langle \text{proof} \rangle$

**lemma** *lemma-st-part-le2*:

$[\![ (x::\text{hypreal}) \in \text{HFinite};$   
 $\text{isLub } \text{Reals} \ \{s. s \in \text{Reals} \ \& \ s < x\} \ t;$   
 $r \in \text{Reals}; \ 0 < r \ ]\!] \implies t + -r \leq x$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-st-part1a*:

$[\![ (x::\text{hypreal}) \in \text{HFinite};$   
 $\text{isLub } \text{Reals} \ \{s. s \in \text{Reals} \ \& \ s < x\} \ t;$   
 $r \in \text{Reals}; \ 0 < r \ ]\!] \implies x + -t \leq r$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-st-part2a*:

$[[ (x::hypreal) \in HFinite;$   
 $isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ t;$   
 $r \in Reals;\ 0 < r ]]$   
 $==> -(x + -t) \leq r$   
 $\langle proof \rangle$

**lemma** *lemma-SReal-ub*:

$(x::hypreal) \in Reals ==> isUb\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ x$   
 $\langle proof \rangle$

**lemma** *lemma-SReal-lub*:

$(x::hypreal) \in Reals ==> isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ x$   
 $\langle proof \rangle$

**lemma** *lemma-st-part-not-eq1*:

$[[ (x::hypreal) \in HFinite;$   
 $isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ t;$   
 $r \in Reals;\ 0 < r ]]$   
 $==> x + -t \neq r$   
 $\langle proof \rangle$

**lemma** *lemma-st-part-not-eq2*:

$[[ (x::hypreal) \in HFinite;$   
 $isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ t;$   
 $r \in Reals;\ 0 < r ]]$   
 $==> -(x + -t) \neq r$   
 $\langle proof \rangle$

**lemma** *lemma-st-part-major*:

$[[ (x::hypreal) \in HFinite;$   
 $isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ t;$   
 $r \in Reals;\ 0 < r ]]$   
 $==> abs\ (x - t) < r$   
 $\langle proof \rangle$

**lemma** *lemma-st-part-major2*:

$[[ (x::hypreal) \in HFinite;\ isLub\ Reals\ \{s.\ s \in Reals\ \&\ s < x\}\ t ]]$   
 $==> \forall r \in Reals.\ 0 < r --> abs\ (x - t) < r$   
 $\langle proof \rangle$

Existence of real and Standard Part Theorem

**lemma** *lemma-st-part-Ex*:

$(x::hypreal) \in HFinite$   
 $==> \exists t \in Reals.\ \forall r \in Reals.\ 0 < r --> abs\ (x - t) < r$   
 $\langle proof \rangle$

**lemma** *st-part-Ex*:

$(x::hypreal) \in HFinite ==> \exists t \in Reals.\ x @= t$

*<proof>*

There is a unique real infinitely close

**lemma** *st-part-Ex1*:  $x \in \mathit{HFinite} \implies \exists! t :: \mathit{hypreal}. t \in \mathit{Reals} \ \& \ x \ @= t$   
*<proof>*

## 8.9 Finite, Infinite and Infinitesimal

**lemma** *HFinite-Int-HInfinite-empty* [*simp*]:  $\mathit{HFinite} \ \mathit{Int} \ \mathit{HInfinite} = \{\}$   
*<proof>*

**lemma** *HFinite-not-HInfinite*:  
**assumes**  $x: x \in \mathit{HFinite}$  **shows**  $x \notin \mathit{HInfinite}$   
*<proof>*

**lemma** *not-HFinite-HInfinite*:  $x \notin \mathit{HFinite} \implies x \in \mathit{HInfinite}$   
*<proof>*

**lemma** *HInfinite-HFinite-disj*:  $x \in \mathit{HInfinite} \mid x \in \mathit{HFinite}$   
*<proof>*

**lemma** *HInfinite-HFinite-iff*:  $(x \in \mathit{HInfinite}) = (x \notin \mathit{HFinite})$   
*<proof>*

**lemma** *HFinite-HInfinite-iff*:  $(x \in \mathit{HFinite}) = (x \notin \mathit{HInfinite})$   
*<proof>*

**lemma** *HInfinite-diff-HFinite-Infinitesimal-disj*:  
 $x \notin \mathit{Infinitesimal} \implies x \in \mathit{HInfinite} \mid x \in \mathit{HFinite} - \mathit{Infinitesimal}$   
*<proof>*

**lemma** *HFinite-inverse*:  
**fixes**  $x :: 'a :: \mathit{real-normed-div-algebra} \ \mathit{star}$   
**shows**  $[\mid x \in \mathit{HFinite}; x \notin \mathit{Infinitesimal} \mid] \implies \mathit{inverse} \ x \in \mathit{HFinite}$   
*<proof>*

**lemma** *HFinite-inverse2*:  
**fixes**  $x :: 'a :: \mathit{real-normed-div-algebra} \ \mathit{star}$   
**shows**  $x \in \mathit{HFinite} - \mathit{Infinitesimal} \implies \mathit{inverse} \ x \in \mathit{HFinite}$   
*<proof>*

**lemma** *Infinitesimal-inverse-HFinite*:  
**fixes**  $x :: 'a :: \mathit{real-normed-div-algebra} \ \mathit{star}$   
**shows**  $x \notin \mathit{Infinitesimal} \implies \mathit{inverse}(x) \in \mathit{HFinite}$   
*<proof>*

**lemma** *HFinite-not-Infinitesimal-inverse*:

**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $x \in \text{HFinite} - \text{Infinitesimal} \implies \text{inverse } x \in \text{HFinite} - \text{Infinitesimal}$   
 $\langle \text{proof} \rangle$

**lemma** *approx-inverse*:  
**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$   
**shows**  
 $\llbracket x \text{@} = y; y \in \text{HFinite} - \text{Infinitesimal} \rrbracket$   
 $\implies \text{inverse } x \text{@} = \text{inverse } y$   
 $\langle \text{proof} \rangle$

**lemmas** *star-of-approx-inverse = star-of-HFinite-diff-Infinitesimal* [THEN [2] *approx-inverse*]  
**lemmas** *hypreal-of-real-approx-inverse = hypreal-of-real-HFinite-diff-Infinitesimal*  
[THEN [2] *approx-inverse*]

**lemma** *inverse-add-Infinitesimal-approx*:  
**fixes**  $x h :: 'a::\text{real-normed-div-algebra star}$   
**shows**  
 $\llbracket x \in \text{HFinite} - \text{Infinitesimal};$   
 $h \in \text{Infinitesimal} \rrbracket \implies \text{inverse}(x + h) \text{@} = \text{inverse } x$   
 $\langle \text{proof} \rangle$

**lemma** *inverse-add-Infinitesimal-approx2*:  
**fixes**  $x h :: 'a::\text{real-normed-div-algebra star}$   
**shows**  
 $\llbracket x \in \text{HFinite} - \text{Infinitesimal};$   
 $h \in \text{Infinitesimal} \rrbracket \implies \text{inverse}(h + x) \text{@} = \text{inverse } x$   
 $\langle \text{proof} \rangle$

**lemma** *inverse-add-Infinitesimal-approx-Infinitesimal*:  
**fixes**  $x h :: 'a::\text{real-normed-div-algebra star}$   
**shows**  
 $\llbracket x \in \text{HFinite} - \text{Infinitesimal};$   
 $h \in \text{Infinitesimal} \rrbracket \implies \text{inverse}(x + h) - \text{inverse } x \text{@} = h$   
 $\langle \text{proof} \rangle$

**lemma** *Infinitesimal-square-iff*:  
**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $(x \in \text{Infinitesimal}) = (x*x \in \text{Infinitesimal})$   
 $\langle \text{proof} \rangle$   
**declare** *Infinitesimal-square-iff* [symmetric, simp]

**lemma** *HFinite-square-iff* [simp]:  
**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $(x*x \in \text{HFinite}) = (x \in \text{HFinite})$   
 $\langle \text{proof} \rangle$

**lemma** *HInfinite-square-iff* [simp]:

**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $(x*x \in HInfinite) = (x \in HInfinite)$   
 $\langle\text{proof}\rangle$

**lemma** *approx-HFinite-mult-cancel:*

**fixes**  $a w z :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $[[ a: HFinite-Infinitesimal; a * w @= a * z ]] ==> w @= z$   
 $\langle\text{proof}\rangle$

**lemma** *approx-HFinite-mult-cancel-iff1:*

**fixes**  $a w z :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $a: HFinite-Infinitesimal ==> (a * w @= a * z) = (w @= z)$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-HFinite-add-cancel:*

$[[ x + y \in HInfinite; y \in HFinite ]] ==> x \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-HFinite-add:*

$[[ x \in HInfinite; y \in HFinite ]] ==> x + y \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-ge-HInfinite:*

$[[ (x::\text{hypreal}) \in HInfinite; x \leq y; 0 \leq x ]] ==> y \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *Infinitesimal-inverse-HInfinite:*

**fixes**  $x :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $[[ x \in Infinitesimal; x \neq 0 ]] ==> \text{inverse } x \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-HFinite-not-Infinitesimal-mult:*

**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $[[ x \in HInfinite; y \in HFinite - Infinitesimal ]]$   
 $==> x * y \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-HFinite-not-Infinitesimal-mult2:*

**fixes**  $x y :: 'a::\text{real-normed-div-algebra star}$   
**shows**  $[[ x \in HInfinite; y \in HFinite - Infinitesimal ]]$   
 $==> y * x \in HInfinite$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-gt-SReal:*

$[[ (x::\text{hypreal}) \in HInfinite; 0 < x; y \in \text{Reals} ]] ==> y < x$   
 $\langle\text{proof}\rangle$

**lemma** *HInfinite-gt-zero-gt-one:*

$[[ (x::\text{hypreal}) \in HInfinite; 0 < x ]] ==> 1 < x$

*<proof>*

**lemma** *not-HInfinite-one* [simp]:  $1 \notin HInfinite$

*<proof>*

**lemma** *approx-hrabs-disj*:  $abs (x::hypreal) @= x \mid abs x @= -x$

*<proof>*

## 8.10 Theorems about Monads

**lemma** *monad-hrabs-Un-subset*:  $monad (abs x) \leq monad(x::hypreal) \cup monad(-x)$

*<proof>*

**lemma** *Infinitesimal-monad-eq*:  $e \in Infinitesimal \implies monad (x+e) = monad x$

*<proof>*

**lemma** *mem-monad-iff*:  $(u \in monad x) = (-u \in monad (-x))$

*<proof>*

**lemma** *Infinitesimal-monad-zero-iff*:  $(x \in Infinitesimal) = (x \in monad 0)$

*<proof>*

**lemma** *monad-zero-minus-iff*:  $(x \in monad 0) = (-x \in monad 0)$

*<proof>*

**lemma** *monad-zero-hrabs-iff*:  $((x::hypreal) \in monad 0) = (abs x \in monad 0)$

*<proof>*

**lemma** *mem-monad-self* [simp]:  $x \in monad x$

*<proof>*

## 8.11 Proof that $x \approx y$ implies $|x| \approx |y|$

**lemma** *approx-subset-monad*:  $x @= y \implies \{x,y\} \leq monad x$

*<proof>*

**lemma** *approx-subset-monad2*:  $x @= y \implies \{x,y\} \leq monad y$

*<proof>*

**lemma** *mem-monad-approx*:  $u \in monad x \implies x @= u$

*<proof>*

**lemma** *approx-mem-monad*:  $x @= u \implies u \in monad x$

*<proof>*

**lemma** *approx-mem-monad2*:  $x @= u \implies x \in monad u$

*<proof>*

**lemma** *approx-mem-monad-zero*:  $[| x @= y; x \in monad 0 |] \implies y \in monad 0$

*<proof>*

**lemma** *Infinitesimal-approx-hrabs:*

$[[ x @= y; (x::hypreal) \in Infinitesimal ]] ==> abs\ x @= abs\ y$   
*<proof>*

**lemma** *less-Infinitesimal-less:*

$[[ 0 < x; (x::hypreal) \notin Infinitesimal; e : Infinitesimal ]] ==> e < x$   
*<proof>*

**lemma** *Ball-mem-monad-gt-zero:*

$[[ 0 < (x::hypreal); x \notin Infinitesimal; u \in monad\ x ]] ==> 0 < u$   
*<proof>*

**lemma** *Ball-mem-monad-less-zero:*

$[[ (x::hypreal) < 0; x \notin Infinitesimal; u \in monad\ x ]] ==> u < 0$   
*<proof>*

**lemma** *lemma-approx-gt-zero:*

$[[ 0 < (x::hypreal); x \notin Infinitesimal; x @= y ]] ==> 0 < y$   
*<proof>*

**lemma** *lemma-approx-less-zero:*

$[[ (x::hypreal) < 0; x \notin Infinitesimal; x @= y ]] ==> y < 0$   
*<proof>*

**theorem** *approx-hrabs:*  $(x::hypreal) @= y ==> abs\ x @= abs\ y$

*<proof>*

**lemma** *approx-hrabs-zero-cancel:*  $abs(x::hypreal) @= 0 ==> x @= 0$

*<proof>*

**lemma** *approx-hrabs-add-Infinitesimal:*

$(e::hypreal) \in Infinitesimal ==> abs\ x @= abs(x+e)$   
*<proof>*

**lemma** *approx-hrabs-add-minus-Infinitesimal:*

$(e::hypreal) \in Infinitesimal ==> abs\ x @= abs(x - e)$   
*<proof>*

**lemma** *hrabs-add-Infinitesimal-cancel:*

$[[ (e::hypreal) \in Infinitesimal; e' \in Infinitesimal; abs(x+e) = abs(y+e') ]] ==> abs\ x @= abs\ y$   
*<proof>*

**lemma** *hrabs-add-minus-Infinitesimal-cancel:*

$[[ (e::hypreal) \in Infinitesimal; e' \in Infinitesimal; abs(x - e) = abs(y - e') ]] ==> abs\ x @= abs\ y$   
*<proof>*

## 8.12 More *HFinite* and *Infinitesimal* Theorems

**lemma** *Infinitesimal-add-hypreal-of-real-less*:

$$\begin{aligned} & [[ x < y; u \in \text{Infinitesimal} ]] \\ & \implies \text{hypreal-of-real } x + u < \text{hypreal-of-real } y \end{aligned}$$

*<proof>*

**lemma** *Infinitesimal-add-hrabs-hypreal-of-real-less*:

$$\begin{aligned} & [[ x \in \text{Infinitesimal}; \text{abs}(\text{hypreal-of-real } r) < \text{hypreal-of-real } y ]] \\ & \implies \text{abs}(\text{hypreal-of-real } r + x) < \text{hypreal-of-real } y \end{aligned}$$

*<proof>*

**lemma** *Infinitesimal-add-hrabs-hypreal-of-real-less2*:

$$\begin{aligned} & [[ x \in \text{Infinitesimal}; \text{abs}(\text{hypreal-of-real } r) < \text{hypreal-of-real } y ]] \\ & \implies \text{abs}(x + \text{hypreal-of-real } r) < \text{hypreal-of-real } y \end{aligned}$$

*<proof>*

**lemma** *hypreal-of-real-le-add-Infinitesimal-cancel*:

$$\begin{aligned} & [[ u \in \text{Infinitesimal}; v \in \text{Infinitesimal}; \\ & \quad \text{hypreal-of-real } x + u \leq \text{hypreal-of-real } y + v ]] \\ & \implies \text{hypreal-of-real } x \leq \text{hypreal-of-real } y \end{aligned}$$

*<proof>*

**lemma** *hypreal-of-real-le-add-Infinitesimal-cancel2*:

$$\begin{aligned} & [[ u \in \text{Infinitesimal}; v \in \text{Infinitesimal}; \\ & \quad \text{hypreal-of-real } x + u \leq \text{hypreal-of-real } y + v ]] \\ & \implies x \leq y \end{aligned}$$

*<proof>*

**lemma** *hypreal-of-real-less-Infinitesimal-le-zero*:

$$[[ \text{hypreal-of-real } x < e; e \in \text{Infinitesimal} ]] \implies \text{hypreal-of-real } x \leq 0$$

*<proof>*

**lemma** *Infinitesimal-add-not-zero*:

$$[[ h \in \text{Infinitesimal}; x \neq 0 ]] \implies \text{star-of } x + h \neq 0$$

*<proof>*

**lemma** *Infinitesimal-square-cancel [simp]*:

$$(x::\text{hypreal}) * x + y * y \in \text{Infinitesimal} \implies x * x \in \text{Infinitesimal}$$

*<proof>*

**lemma** *HFinite-square-cancel [simp]*:

$$(x::\text{hypreal}) * x + y * y \in \text{HFinite} \implies x * x \in \text{HFinite}$$

*<proof>*

**lemma** *Infinitesimal-square-cancel2 [simp]*:

$$(x::\text{hypreal}) * x + y * y \in \text{Infinitesimal} \implies y * y \in \text{Infinitesimal}$$

*<proof>*

**lemma** *HFinite-square-cancel2* [simp]:  
 $(x::\text{hypreal}) * x + y * y \in \text{HFinite} \implies y * y \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *Infinitesimal-sum-square-cancel* [simp]:  
 $(x::\text{hypreal}) * x + y * y + z * z \in \text{Infinitesimal} \implies x * x \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *HFinite-sum-square-cancel* [simp]:  
 $(x::\text{hypreal}) * x + y * y + z * z \in \text{HFinite} \implies x * x \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *Infinitesimal-sum-square-cancel2* [simp]:  
 $(y::\text{hypreal}) * y + x * x + z * z \in \text{Infinitesimal} \implies x * x \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *HFinite-sum-square-cancel2* [simp]:  
 $(y::\text{hypreal}) * y + x * x + z * z \in \text{HFinite} \implies x * x \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *Infinitesimal-sum-square-cancel3* [simp]:  
 $(z::\text{hypreal}) * z + y * y + x * x \in \text{Infinitesimal} \implies x * x \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *HFinite-sum-square-cancel3* [simp]:  
 $(z::\text{hypreal}) * z + y * y + x * x \in \text{HFinite} \implies x * x \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *monad-hrabs-less*:  
 $[[ y \in \text{monad } x; 0 < \text{hypreal-of-real } e ]]$   
 $\implies \text{abs } (y - x) < \text{hypreal-of-real } e$   
 ⟨proof⟩

**lemma** *mem-monad-SReal-HFinite*:  
 $x \in \text{monad } (\text{hypreal-of-real } a) \implies x \in \text{HFinite}$   
 ⟨proof⟩

### 8.13 Theorems about Standard Part

**lemma** *st-approx-self*:  $x \in \text{HFinite} \implies \text{st } x @ = x$   
 ⟨proof⟩

**lemma** *st-SReal*:  $x \in \text{HFinite} \implies \text{st } x \in \text{Reals}$   
 ⟨proof⟩

**lemma** *st-HFinite*:  $x \in \text{HFinite} \implies \text{st } x \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *st-unique*:  $[[ r \in \mathbb{R}; r \approx x ]]$   $\implies \text{st } x = r$

*<proof>*

**lemma** *st-SReal-eq*:  $x \in \text{Reals} \implies st\ x = x$   
*<proof>*

**lemma** *st-hypreal-of-real [simp]*:  $st\ (\text{hypreal-of-real } x) = \text{hypreal-of-real } x$   
*<proof>*

**lemma** *st-eq-approx*:  $[[\ x \in \text{HFinite};\ y \in \text{HFinite};\ st\ x = st\ y\ ]]$   $\implies x \text{ @=} y$   
*<proof>*

**lemma** *approx-st-eq*:  
**assumes**  $x: x \in \text{HFinite}$  **and**  $y: y \in \text{HFinite}$  **and**  $xy: x \text{ @=} y$   
**shows**  $st\ x = st\ y$   
*<proof>*

**lemma** *st-eq-approx-iff*:  
 $[[\ x \in \text{HFinite};\ y \in \text{HFinite}]]$   
 $\implies (x \text{ @=} y) = (st\ x = st\ y)$   
*<proof>*

**lemma** *st-Infinitesimal-add-SReal*:  
 $[[\ x \in \text{Reals};\ e \in \text{Infinitesimal} ]]$   $\implies st(x + e) = x$   
*<proof>*

**lemma** *st-Infinitesimal-add-SReal2*:  
 $[[\ x \in \text{Reals};\ e \in \text{Infinitesimal} ]]$   $\implies st(e + x) = x$   
*<proof>*

**lemma** *HFinite-st-Infinitesimal-add*:  
 $x \in \text{HFinite} \implies \exists e \in \text{Infinitesimal}. x = st(x) + e$   
*<proof>*

**lemma** *st-add*:  $[[\ x \in \text{HFinite};\ y \in \text{HFinite}]] \implies st(x + y) = st\ x + st\ y$   
*<proof>*

**lemma** *st-numeral [simp]*:  $st\ (\text{numeral } w) = \text{numeral } w$   
*<proof>*

**lemma** *st-neg-numeral [simp]*:  $st\ (\text{neg-numeral } w) = \text{neg-numeral } w$   
*<proof>*

**lemma** *st-0 [simp]*:  $st\ 0 = 0$   
*<proof>*

**lemma** *st-1 [simp]*:  $st\ 1 = 1$   
*<proof>*

**lemma** *st-minus*:  $x \in \text{HFinite} \implies st(-x) = -st\ x$

$\langle proof \rangle$

**lemma** *st-diff*:  $\llbracket x \in HFinite; y \in HFinite \rrbracket \implies st(x - y) = st x - st y$   
 $\langle proof \rangle$

**lemma** *st-mult*:  $\llbracket x \in HFinite; y \in HFinite \rrbracket \implies st(x * y) = st x * st y$   
 $\langle proof \rangle$

**lemma** *st-Infinitesimal*:  $x \in Infinitesimal \implies st x = 0$   
 $\langle proof \rangle$

**lemma** *st-not-Infinitesimal*:  $st(x) \neq 0 \implies x \notin Infinitesimal$   
 $\langle proof \rangle$

**lemma** *st-inverse*:  
 $\llbracket x \in HFinite; st x \neq 0 \rrbracket$   
 $\implies st(inverse x) = inverse(st x)$   
 $\langle proof \rangle$

**lemma** *st-divide* [*simp*]:  
 $\llbracket x \in HFinite; y \in HFinite; st y \neq 0 \rrbracket$   
 $\implies st(x/y) = (st x) / (st y)$   
 $\langle proof \rangle$

**lemma** *st-idempotent* [*simp*]:  $x \in HFinite \implies st(st(x)) = st(x)$   
 $\langle proof \rangle$

**lemma** *Infinitesimal-add-st-less*:  
 $\llbracket x \in HFinite; y \in HFinite; u \in Infinitesimal; st x < st y \rrbracket$   
 $\implies st x + u < st y$   
 $\langle proof \rangle$

**lemma** *Infinitesimal-add-st-le-cancel*:  
 $\llbracket x \in HFinite; y \in HFinite;$   
 $u \in Infinitesimal; st x \leq st y + u$   
 $\rrbracket \implies st x \leq st y$   
 $\langle proof \rangle$

**lemma** *st-le*:  $\llbracket x \in HFinite; y \in HFinite; x \leq y \rrbracket \implies st(x) \leq st(y)$   
 $\langle proof \rangle$

**lemma** *st-zero-le*:  $\llbracket 0 \leq x; x \in HFinite \rrbracket \implies 0 \leq st x$   
 $\langle proof \rangle$

**lemma** *st-zero-ge*:  $\llbracket x \leq 0; x \in HFinite \rrbracket \implies st x \leq 0$   
 $\langle proof \rangle$

**lemma** *st-hrabs*:  $x \in HFinite \implies abs(st x) = st(abs x)$   
 $\langle proof \rangle$

## 8.14 Alternative Definitions using Free Ultrafilter

### 8.14.1 *HFinite*

**lemma** *HFinite-FreeUltrafilterNat*:

$$\text{star-}n X \in \text{HFinite}$$

$$\implies \exists u. \{n. \text{norm } (X n) < u\} \in \text{FreeUltrafilterNat}$$

*<proof>*

**lemma** *FreeUltrafilterNat-HFinite*:

$$\exists u. \{n. \text{norm } (X n) < u\} \in \text{FreeUltrafilterNat}$$

$$\implies \text{star-}n X \in \text{HFinite}$$

*<proof>*

**lemma** *HFinite-FreeUltrafilterNat-iff*:

$$(\text{star-}n X \in \text{HFinite}) = (\exists u. \{n. \text{norm } (X n) < u\} \in \text{FreeUltrafilterNat})$$

*<proof>*

### 8.14.2 *HInfinite*

**lemma** *lemma-Compl-eq*:  $-\{n. u < \text{norm } (xa n)\} = \{n. \text{norm } (xa n) \leq u\}$

*<proof>*

**lemma** *lemma-Compl-eq2*:  $-\{n. \text{norm } (xa n) < u\} = \{n. u \leq \text{norm } (xa n)\}$

*<proof>*

**lemma** *lemma-Int-eq1*:

$$\{n. \text{norm } (xa n) \leq u\} \text{ Int } \{n. u \leq \text{norm } (xa n)\}$$

$$= \{n. \text{norm } (xa n) = u\}$$

*<proof>*

**lemma** *lemma-FreeUltrafilterNat-one*:

$$\{n. \text{norm } (xa n) = u\} \leq \{n. \text{norm } (xa n) < u + (1::\text{real})\}$$

*<proof>*

**lemma** *FreeUltrafilterNat-const-Finite*:

$$\{n. \text{norm } (X n) = u\} \in \text{FreeUltrafilterNat} \implies \text{star-}n X \in \text{HFinite}$$

*<proof>*

**lemma** *HInfinite-FreeUltrafilterNat*:

$$\text{star-}n X \in \text{HInfinite} \implies \forall u. \{n. u < \text{norm } (X n)\} \in \text{FreeUltrafilterNat}$$

*<proof>*

**lemma** *lemma-Int-HI*:

$$\{n. \text{norm } (Xa n) < u\} \text{ Int } \{n. X n = Xa n\} \subseteq \{n. \text{norm } (X n) < (u::\text{real})\}$$

*<proof>*

**lemma** *lemma-Int-HIa*:  $\{n. u < \text{norm } (X n)\} \text{ Int } \{n. \text{norm } (X n) < u\} = \{\}$

*<proof>*

**lemma** *FreeUltrafilterNat-HInfinite*:

$\forall u. \{n. u < \text{norm } (X\ n)\} \in \text{FreeUltrafilterNat} \implies \text{star-}n\ X \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *HInfinite-FreeUltrafilterNat-iff*:

$(\text{star-}n\ X \in \text{HInfinite}) = (\forall u. \{n. u < \text{norm } (X\ n)\} \in \text{FreeUltrafilterNat})$   
 ⟨proof⟩

### 8.14.3 Infinitesimal

**lemma** *ball-SReal-eq*:  $(\forall x::\text{hypreal} \in \text{Reals}. P\ x) = (\forall x::\text{real}. P\ (\text{star-of } x))$   
 ⟨proof⟩

**lemma** *Infinitesimal-FreeUltrafilterNat*:

$\text{star-}n\ X \in \text{Infinitesimal} \implies \forall u > 0. \{n. \text{norm } (X\ n) < u\} \in \mathcal{U}$   
 ⟨proof⟩

**lemma** *FreeUltrafilterNat-Infinitesimal*:

$\forall u > 0. \{n. \text{norm } (X\ n) < u\} \in \mathcal{U} \implies \text{star-}n\ X \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *Infinitesimal-FreeUltrafilterNat-iff*:

$(\text{star-}n\ X \in \text{Infinitesimal}) = (\forall u > 0. \{n. \text{norm } (X\ n) < u\} \in \mathcal{U})$   
 ⟨proof⟩

**lemma** *lemma-Infinitesimal*:

$(\forall r. 0 < r \implies x < r) = (\forall n. x < \text{inverse}(\text{real } (\text{Suc } n)))$   
 ⟨proof⟩

**lemma** *lemma-Infinitesimal2*:

$(\forall r \in \text{Reals}. 0 < r \implies x < r) =$   
 $(\forall n. x < \text{inverse}(\text{hypreal-of-nat } (\text{Suc } n)))$   
 ⟨proof⟩

**lemma** *Infinitesimal-hypreal-of-nat-iff*:

$\text{Infinitesimal} = \{x. \forall n. \text{hnorm } x < \text{inverse } (\text{hypreal-of-nat } (\text{Suc } n))\}$   
 ⟨proof⟩

## 8.15 Proof that $\omega$ is an infinite number

It will follow that epsilon is an infinitesimal number.

**lemma** *Suc-Un-eq*:  $\{n. n < \text{Suc } m\} = \{n. n < m\} \cup \{n. n = m\}$   
 ⟨proof⟩

**lemma** *finite-nat-segment*:  $\text{finite } \{n::\text{nat}. n < m\}$   
 ⟨proof⟩

**lemma** *finite-real-of-nat-segment*:  $\text{finite } \{n::\text{nat}. \text{real } n < \text{real } (m::\text{nat})\}$   
 ⟨proof⟩

**lemma** *finite-real-of-nat-less-real*:  $\text{finite } \{n::\text{nat}. \text{real } n < u\}$   
 ⟨proof⟩

**lemma** *lemma-real-le-Un-eq*:  
 $\{n. f n \leq u\} = \{n. f n < u\} \cup \{n. u = (f n :: \text{real})\}$   
 ⟨proof⟩

**lemma** *finite-real-of-nat-le-real*:  $\text{finite } \{n::\text{nat}. \text{real } n \leq u\}$   
 ⟨proof⟩

**lemma** *finite-rabs-real-of-nat-le-real*:  $\text{finite } \{n::\text{nat}. \text{abs}(\text{real } n) \leq u\}$   
 ⟨proof⟩

**lemma** *rabs-real-of-nat-le-real-FreeUltrafilterNat*:  
 $\{n. \text{abs}(\text{real } n) \leq u\} \notin \text{FreeUltrafilterNat}$   
 ⟨proof⟩

**lemma** *FreeUltrafilterNat-nat-gt-real*:  $\{n. u < \text{real } n\} \in \text{FreeUltrafilterNat}$   
 ⟨proof⟩

**lemma** *Compl-real-le-eq*:  $\neg \{n::\text{nat}. \text{real } n \leq u\} = \{n. u < \text{real } n\}$   
 ⟨proof⟩

$\omega$  is a member of *HInfinite*

**lemma** *FreeUltrafilterNat-omega*:  $\{n. u < \text{real } n\} \in \text{FreeUltrafilterNat}$   
 ⟨proof⟩

**theorem** *HInfinite-omega [simp]*:  $\omega \in \text{HInfinite}$   
 ⟨proof⟩

**lemma** *Infinitesimal-epsilon [simp]*:  $\epsilon \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *HFinite-epsilon [simp]*:  $\epsilon \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *epsilon-approx-zero [simp]*:  $\epsilon @= 0$   
 ⟨proof⟩

**lemma** *real-of-nat-less-inverse-iff*:

$$0 < u \implies (u < \text{inverse}(\text{real}(\text{Suc } n))) = (\text{real}(\text{Suc } n) < \text{inverse } u)$$

*<proof>*

**lemma** *finite-inverse-real-of-posnat-gt-real*:

$$0 < u \implies \text{finite } \{n. u < \text{inverse}(\text{real}(\text{Suc } n))\}$$

*<proof>*

**lemma** *lemma-real-le-Un-eq2*:

$$\begin{aligned} & \{n. u \leq \text{inverse}(\text{real}(\text{Suc } n))\} = \\ & \{n. u < \text{inverse}(\text{real}(\text{Suc } n))\} \text{ Un } \{n. u = \text{inverse}(\text{real}(\text{Suc } n))\} \end{aligned}$$

*<proof>*

**lemma** *real-of-nat-inverse-eq-iff*:

$$(u = \text{inverse}(\text{real}(\text{Suc } n))) = (\text{real}(\text{Suc } n) = \text{inverse } u)$$

*<proof>*

**lemma** *lemma-finite-omega-set2*:  $\text{finite } \{n::\text{nat}. u = \text{inverse}(\text{real}(\text{Suc } n))\}$

*<proof>*

**lemma** *finite-inverse-real-of-posnat-ge-real*:

$$0 < u \implies \text{finite } \{n. u \leq \text{inverse}(\text{real}(\text{Suc } n))\}$$

*<proof>*

**lemma** *inverse-real-of-posnat-ge-real-FreeUltrafilterNat*:

$$0 < u \implies \{n. u \leq \text{inverse}(\text{real}(\text{Suc } n))\} \notin \text{FreeUltrafilterNat}$$

*<proof>*

**lemma** *Compl-le-inverse-eq*:

$$\begin{aligned} & - \{n. u \leq \text{inverse}(\text{real}(\text{Suc } n))\} = \\ & \{n. \text{inverse}(\text{real}(\text{Suc } n)) < u\} \end{aligned}$$

*<proof>*

**lemma** *FreeUltrafilterNat-inverse-real-of-posnat*:

$$0 < u \implies \{n. \text{inverse}(\text{real}(\text{Suc } n)) < u\} \in \text{FreeUltrafilterNat}$$

*<proof>*

Example of an hypersequence (i.e. an extended standard sequence) whose term with an hypernatural suffix is an infinitesimal i.e. the  $\text{whn}'\text{nth}$  term of the hypersequence is a member of *Infinitesimal*

**lemma** *SEQ-Infinitesimal*:

$$(*f* (\%n::\text{nat}. \text{inverse}(\text{real}(\text{Suc } n)))) \text{ whn} : \text{Infinitesimal}$$

*<proof>*

Example where we get a hyperreal from a real sequence for which a par-

ticular property holds. The theorem is used in proofs about equivalence of nonstandard and standard neighbourhoods. Also used for equivalence of nonstandard and standard definitions of pointwise limit.

**lemma** *real-seq-to-hypreal-Infinitesimal*:

$$\forall n. \text{norm}(X\ n - x) < \text{inverse}(\text{real}(\text{Suc}\ n)) \\ \implies \text{star-n}\ X - \text{star-of}\ x \in \text{Infinitesimal}$$

*<proof>*

**lemma** *real-seq-to-hypreal-approx*:

$$\forall n. \text{norm}(X\ n - x) < \text{inverse}(\text{real}(\text{Suc}\ n)) \\ \implies \text{star-n}\ X \ @= \text{star-of}\ x$$

*<proof>*

**lemma** *real-seq-to-hypreal-approx2*:

$$\forall n. \text{norm}(x - X\ n) < \text{inverse}(\text{real}(\text{Suc}\ n)) \\ \implies \text{star-n}\ X \ @= \text{star-of}\ x$$

*<proof>*

**lemma** *real-seq-to-hypreal-Infinitesimal2*:

$$\forall n. \text{norm}(X\ n - Y\ n) < \text{inverse}(\text{real}(\text{Suc}\ n)) \\ \implies \text{star-n}\ X - \text{star-n}\ Y \in \text{Infinitesimal}$$

*<proof>*

**end**

## 9 NSComplex: Nonstandard Complex Numbers

**theory** *NSComplex*

**imports** *Complex NSA*

**begin**

**type-synonym** *hcomplex = complex star*

**abbreviation**

*hcomplex-of-complex* :: *complex => complex star* **where**  
*hcomplex-of-complex* == *star-of*

**abbreviation**

*hmod* :: *complex star => real star* **where**  
*hmod* == *hnorm*

**definition**

*hRe* :: *hcomplex => hypreal* **where**  
*hRe* = *\*f\* Re*

**definition**

$hIm :: hcomplex \Rightarrow hypreal$  **where**  
 $hIm = *f* Im$

**definition**

$iii :: hcomplex$  **where**  
 $iii = star-of ii$

**definition**

$hcnj :: hcomplex \Rightarrow hcomplex$  **where**  
 $hcnj = *f* cnj$

**definition**

$hsgn :: hcomplex \Rightarrow hcomplex$  **where**  
 $hsgn = *f* sgn$

**definition**

$harg :: hcomplex \Rightarrow hypreal$  **where**  
 $harg = *f* arg$

**definition**

$hcis :: hypreal \Rightarrow hcomplex$  **where**  
 $hcis = *f* cis$

**abbreviation**

$hcomplex-of-hypreal :: hypreal \Rightarrow hcomplex$  **where**  
 $hcomplex-of-hypreal \equiv of-hypreal$

**definition**

$hrcis :: [hypreal, hypreal] \Rightarrow hcomplex$  **where**  
 $hrcis = *f2* rcis$

**definition**

$hexpi :: hcomplex \Rightarrow hcomplex$  **where**  
 $hexpi = *f* expi$

**definition**

$HComplex :: [hypreal, hypreal] => hcomplex$  **where**  
 $HComplex = *f2* Complex$

**lemmas**  $hcomplex-defs$  [transfer-unfold] =  
 $hRe-def$   $hIm-def$   $iii-def$   $hcnj-def$   $hsgn-def$   $harg-def$   $hcis-def$   
 $hrcis-def$   $hexpi-def$   $HComplex-def$

**lemma**  $Standard-hRe$  [simp]:  $x \in Standard \implies hRe\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hIm$  [simp]:  $x \in Standard \implies hIm\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-iii$  [simp]:  $iii \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hcnj$  [simp]:  $x \in Standard \implies hcnj\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hsgn$  [simp]:  $x \in Standard \implies hsgn\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-harg$  [simp]:  $x \in Standard \implies harg\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hcis$  [simp]:  $r \in Standard \implies hcis\ r \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hexpi$  [simp]:  $x \in Standard \implies hexpi\ x \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-hrcis$  [simp]:  
 $\llbracket r \in Standard; s \in Standard \rrbracket \implies hrcis\ r\ s \in Standard$   
 ⟨proof⟩

**lemma**  $Standard-HComplex$  [simp]:  
 $\llbracket r \in Standard; s \in Standard \rrbracket \implies HComplex\ r\ s \in Standard$   
 ⟨proof⟩

**lemma**  $hcmmod-def$ :  $hcmmod = *f* cmod$   
 ⟨proof⟩

**9.1 Properties of Nonstandard Real and Imaginary Parts**

**lemma**  $hcomplex-hRe-hIm-cancel-iff$ :  
 $\llbracket w\ z. (w=z) = (hRe(w) = hRe(z) \ \& \ hIm(w) = hIm(z)) \rrbracket$   
 ⟨proof⟩

**lemma** *hcomplex-equality* [*intro?*]:  
 $!!z w. \text{hRe } z = \text{hRe } w \implies \text{hIm } z = \text{hIm } w \implies z = w$   
 ⟨*proof*⟩

**lemma** *hcomplex-hRe-zero* [*simp*]:  $\text{hRe } 0 = 0$   
 ⟨*proof*⟩

**lemma** *hcomplex-hIm-zero* [*simp*]:  $\text{hIm } 0 = 0$   
 ⟨*proof*⟩

**lemma** *hcomplex-hRe-one* [*simp*]:  $\text{hRe } 1 = 1$   
 ⟨*proof*⟩

**lemma** *hcomplex-hIm-one* [*simp*]:  $\text{hIm } 1 = 0$   
 ⟨*proof*⟩

## 9.2 Addition for Nonstandard Complex Numbers

**lemma** *hRe-add*:  $!!x y. \text{hRe}(x + y) = \text{hRe}(x) + \text{hRe}(y)$   
 ⟨*proof*⟩

**lemma** *hIm-add*:  $!!x y. \text{hIm}(x + y) = \text{hIm}(x) + \text{hIm}(y)$   
 ⟨*proof*⟩

## 9.3 More Minus Laws

**lemma** *hRe-minus*:  $!!z. \text{hRe}(-z) = -\text{hRe}(z)$   
 ⟨*proof*⟩

**lemma** *hIm-minus*:  $!!z. \text{hIm}(-z) = -\text{hIm}(z)$   
 ⟨*proof*⟩

**lemma** *hcomplex-add-minus-eq-minus*:  
 $x + y = (0::\text{hcomplex}) \implies x = -y$   
 ⟨*proof*⟩

**lemma** *hcomplex-i-mult-eq* [*simp*]:  $iii * iii = -1$   
 ⟨*proof*⟩

**lemma** *hcomplex-i-mult-left* [*simp*]:  $!!z. iii * (iii * z) = -z$   
 ⟨*proof*⟩

**lemma** *hcomplex-i-not-zero* [*simp*]:  $iii \neq 0$   
 ⟨*proof*⟩

## 9.4 More Multiplication Laws

**lemma** *hcomplex-mult-minus-one*:  $-1 * (z::\text{hcomplex}) = -z$   
 ⟨*proof*⟩

**lemma** *hcomplex-mult-minus-one-right*:  $(z::hcomplex) * - 1 = -z$   
 ⟨proof⟩

**lemma** *hcomplex-mult-left-cancel*:  
 $(c::hcomplex) \neq (0::hcomplex) \implies (c*a=c*b) = (a=b)$   
 ⟨proof⟩

**lemma** *hcomplex-mult-right-cancel*:  
 $(c::hcomplex) \neq (0::hcomplex) \implies (a*c=b*c) = (a=b)$   
 ⟨proof⟩

## 9.5 Subraction and Division

**lemma** *hcomplex-diff-eq-eq* [simp]:  $((x::hcomplex) - y = z) = (x = z + y)$   
 ⟨proof⟩

## 9.6 Embedding Properties for *hcomplex-of-hypreal* Map

**lemma** *hRe-hcomplex-of-hypreal* [simp]:  $!!z. hRe(hcomplex-of-hypreal z) = z$   
 ⟨proof⟩

**lemma** *hIm-hcomplex-of-hypreal* [simp]:  $!!z. hIm(hcomplex-of-hypreal z) = 0$   
 ⟨proof⟩

**lemma** *hcomplex-of-hypreal-epsilon-not-zero* [simp]:  
 $hcomplex-of-hypreal\ epsilon \neq 0$   
 ⟨proof⟩

## 9.7 HComplex theorems

**lemma** *hRe-HComplex* [simp]:  $!!x\ y. hRe (HComplex\ x\ y) = x$   
 ⟨proof⟩

**lemma** *hIm-HComplex* [simp]:  $!!x\ y. hIm (HComplex\ x\ y) = y$   
 ⟨proof⟩

**lemma** *hcomplex-surj* [simp]:  $!!z. HComplex (hRe z) (hIm z) = z$   
 ⟨proof⟩

**lemma** *hcomplex-induct* [case-names rect]:  
 $(\bigwedge x\ y. P (HComplex\ x\ y)) \implies P z$   
 ⟨proof⟩

## 9.8 Modulus (Absolute Value) of Nonstandard Complex Number

**lemma** *hcomplex-of-hypreal-abs*:  
 $hcomplex-of-hypreal (abs\ x) =$   
 $hcomplex-of-hypreal(hcmod(hcomplex-of-hypreal\ x))$

*<proof>*

**lemma** *HComplex-inject* [simp]:

$!!x\ y\ x'\ y'.\ HComplex\ x\ y = HComplex\ x'\ y' = (x=x' \ \&\ y=y')$

*<proof>*

**lemma** *HComplex-add* [simp]:

$!!x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 + HComplex\ x2\ y2 = HComplex\ (x1+x2)\ (y1+y2)$

*<proof>*

**lemma** *HComplex-minus* [simp]:  $!!x\ y.\ -\ HComplex\ x\ y = HComplex\ (-x)\ (-y)$

*<proof>*

**lemma** *HComplex-diff* [simp]:

$!!x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 - HComplex\ x2\ y2 = HComplex\ (x1-x2)\ (y1-y2)$

*<proof>*

**lemma** *HComplex-mult* [simp]:

$!!x1\ y1\ x2\ y2.\ HComplex\ x1\ y1 * HComplex\ x2\ y2 =$   
 $HComplex\ (x1*x2 - y1*y2)\ (x1*y2 + y1*x2)$

*<proof>*

**lemma** *hcomplex-of-hypreal-eq*:  $!!r.\ hcomplex-of-hypreal\ r = HComplex\ r\ 0$

*<proof>*

**lemma** *HComplex-add-hcomplex-of-hypreal* [simp]:

$!!x\ y\ r.\ HComplex\ x\ y + hcomplex-of-hypreal\ r = HComplex\ (x+r)\ y$

*<proof>*

**lemma** *hcomplex-of-hypreal-add-HComplex* [simp]:

$!!r\ x\ y.\ hcomplex-of-hypreal\ r + HComplex\ x\ y = HComplex\ (r+x)\ y$

*<proof>*

**lemma** *HComplex-mult-hcomplex-of-hypreal*:

$!!x\ y\ r.\ HComplex\ x\ y * hcomplex-of-hypreal\ r = HComplex\ (x*r)\ (y*r)$

*<proof>*

**lemma** *hcomplex-of-hypreal-mult-HComplex*:

$!!r\ x\ y.\ hcomplex-of-hypreal\ r * HComplex\ x\ y = HComplex\ (r*x)\ (r*y)$

*<proof>*

**lemma** *i-hcomplex-of-hypreal* [simp]:

$!!r.\ iii * hcomplex-of-hypreal\ r = HComplex\ 0\ r$

*<proof>*

**lemma** *hcomplex-of-hypreal-i* [simp]:

$!!r.\ hcomplex-of-hypreal\ r * iii = HComplex\ 0\ r$

*<proof>*

## 9.9 Conjugation

**lemma** *hcomplex-hcnj-cancel-iff* [*iff*]:  $\forall x y. (hcnj\ x = hcnj\ y) = (x = y)$   
*<proof>*

**lemma** *hcomplex-hcnj-hcnj* [*simp*]:  $\forall z. hcnj\ (hcnj\ z) = z$   
*<proof>*

**lemma** *hcomplex-hcnj-hcomplex-of-hypreal* [*simp*]:  
 $\forall x. hcnj\ (hcomplex-of-hypreal\ x) = hcomplex-of-hypreal\ x$   
*<proof>*

**lemma** *hcomplex-hmod-hcnj* [*simp*]:  $\forall z. hmod\ (hcnj\ z) = hmod\ z$   
*<proof>*

**lemma** *hcomplex-hcnj-minus*:  $\forall z. hcnj\ (-z) = -\ hcnj\ z$   
*<proof>*

**lemma** *hcomplex-hcnj-inverse*:  $\forall z. hcnj\ (inverse\ z) = inverse\ (hcnj\ z)$   
*<proof>*

**lemma** *hcomplex-hcnj-add*:  $\forall w\ z. hcnj\ (w + z) = hcnj\ w + hcnj\ z$   
*<proof>*

**lemma** *hcomplex-hcnj-diff*:  $\forall w\ z. hcnj\ (w - z) = hcnj\ w - hcnj\ z$   
*<proof>*

**lemma** *hcomplex-hcnj-mult*:  $\forall w\ z. hcnj\ (w * z) = hcnj\ w * hcnj\ z$   
*<proof>*

**lemma** *hcomplex-hcnj-divide*:  $\forall w\ z. hcnj\ (w / z) = (hcnj\ w) / (hcnj\ z)$   
*<proof>*

**lemma** *hcnj-one* [*simp*]:  $hcnj\ 1 = 1$   
*<proof>*

**lemma** *hcomplex-hcnj-zero* [*simp*]:  $hcnj\ 0 = 0$   
*<proof>*

**lemma** *hcomplex-hcnj-zero-iff* [*iff*]:  $\forall z. (hcnj\ z = 0) = (z = 0)$   
*<proof>*

**lemma** *hcomplex-mult-hcnj*:  
 $\forall z. z * hcnj\ z = hcomplex-of-hypreal\ (hRe(z) ^ 2 + hIm(z) ^ 2)$   
*<proof>*

### 9.10 More Theorems about the Function $hcm\text{od}$

**lemma**  $hcm\text{od-}h\text{complex-}of\text{-}hyp\text{real-}of\text{-}nat$  [simp]:  
 $hcm\text{od} (h\text{complex-}of\text{-}hyp\text{real}(hyp\text{real-}of\text{-}nat\ n)) = hyp\text{real-}of\text{-}nat\ n$   
 ⟨proof⟩

**lemma**  $hcm\text{od-}h\text{complex-}of\text{-}hyp\text{real-}of\text{-}hyp\text{nat}$  [simp]:  
 $hcm\text{od} (h\text{complex-}of\text{-}hyp\text{real}(hyp\text{real-}of\text{-}hyp\text{nat}\ n)) = hyp\text{real-}of\text{-}hyp\text{nat}\ n$   
 ⟨proof⟩

**lemma**  $hcm\text{od-}mult\text{-}hcnj$ :  $!!z. hcm\text{od}(z * hcnj(z)) = hcm\text{od}(z) ^ 2$   
 ⟨proof⟩

**lemma**  $hcm\text{od-}triangle\text{-}ineq2$  [simp]:  
 $!!a\ b. hcm\text{od}(b + a) - hcm\text{od}\ b \leq hcm\text{od}\ a$   
 ⟨proof⟩

**lemma**  $hcm\text{od-}diff\text{-}ineq$  [simp]:  $!!a\ b. hcm\text{od}(a) - hcm\text{od}(b) \leq hcm\text{od}(a + b)$   
 ⟨proof⟩

### 9.11 Exponentiation

**lemma**  $h\text{complex}pow\text{-}0$  [simp]:  $z ^ 0 = (1::h\text{complex})$   
 ⟨proof⟩

**lemma**  $h\text{complex}pow\text{-}Suc$  [simp]:  $z ^ (Suc\ n) = (z::h\text{complex}) * (z ^ n)$   
 ⟨proof⟩

**lemma**  $h\text{complex}pow\text{-}i\text{-}squared$  [simp]:  $iii ^ 2 = -1$   
 ⟨proof⟩

**lemma**  $h\text{complex-}of\text{-}hyp\text{real-}pow$ :  
 $!!x. h\text{complex-}of\text{-}hyp\text{real} (x ^ n) = (h\text{complex-}of\text{-}hyp\text{real}\ x) ^ n$   
 ⟨proof⟩

**lemma**  $h\text{complex-}hcnj\text{-}pow$ :  $!!z. hcnj(z ^ n) = hcnj(z) ^ n$   
 ⟨proof⟩

**lemma**  $hcm\text{od-}h\text{complex}pow$ :  $!!x. hcm\text{od}(x ^ n) = hcm\text{od}(x) ^ n$   
 ⟨proof⟩

**lemma**  $h\text{cpow-}minus$ :  
 $!!x\ n. (-x::h\text{complex})\ pow\ n =$   
 $(if\ (*p*\ even)\ n\ then\ (x\ pow\ n)\ else\ -(x\ pow\ n))$   
 ⟨proof⟩

**lemma**  $h\text{cpow-}mult$ :  
 $!!r\ s\ n. ((r::h\text{complex}) * s)\ pow\ n = (r\ pow\ n) * (s\ pow\ n)$   
 ⟨proof⟩

**lemma** *hcpow-zero2* [*simp*]:  
 $\bigwedge n. 0 \text{ pow } (hSuc\ n) = (0::'a::\{\text{power, semiring-0}\}\ \text{star})$   
 ⟨*proof*⟩

**lemma** *hcpow-not-zero* [*simp, intro*]:  
 $!!r\ n. r \neq 0 \implies r \text{ pow } n \neq (0::hcomplex)$   
 ⟨*proof*⟩

**lemma** *hcpow-zero-zero*:  $r \text{ pow } n = (0::hcomplex) \implies r = 0$   
 ⟨*proof*⟩

## 9.12 The Function *hsgn*

**lemma** *hsgn-zero* [*simp*]:  $hsgn\ 0 = 0$   
 ⟨*proof*⟩

**lemma** *hsgn-one* [*simp*]:  $hsgn\ 1 = 1$   
 ⟨*proof*⟩

**lemma** *hsgn-minus*:  $!!z. hsgn\ (-z) = -\ hsgn(z)$   
 ⟨*proof*⟩

**lemma** *hsgn-eq*:  $!!z. hsgn\ z = z / hcomplex\ of\ hypreal\ (hcm\ mod\ z)$   
 ⟨*proof*⟩

**lemma** *hcm\ mod\ -i*:  $!!x\ y. hcm\ mod\ (HComplex\ x\ y) = (*f*\ sqrt)\ (x\ ^\ 2\ +\ y\ ^\ 2)$   
 ⟨*proof*⟩

**lemma** *hcomplex-eq-cancel-iff1* [*simp*]:  
 $(hcomplex\ of\ hypreal\ xa = HComplex\ x\ y) = (x\ a = x\ \&\ y = 0)$   
 ⟨*proof*⟩

**lemma** *hcomplex-eq-cancel-iff2* [*simp*]:  
 $(HComplex\ x\ y = hcomplex\ of\ hypreal\ xa) = (x = x\ a\ \&\ y = 0)$   
 ⟨*proof*⟩

**lemma** *HComplex-eq-0* [*simp*]:  $!!x\ y. (HComplex\ x\ y = 0) = (x = 0\ \&\ y = 0)$   
 ⟨*proof*⟩

**lemma** *HComplex-eq-1* [*simp*]:  $!!x\ y. (HComplex\ x\ y = 1) = (x = 1\ \&\ y = 0)$   
 ⟨*proof*⟩

**lemma** *i-eq-HComplex-0-1*:  $iii = HComplex\ 0\ 1$   
 ⟨*proof*⟩

**lemma** *HComplex-eq-i* [*simp*]:  $!!x\ y. (HComplex\ x\ y = iii) = (x = 0\ \&\ y = 1)$   
 ⟨*proof*⟩

**lemma** *hRe-hsgn* [*simp*]:  $!!z. hRe(hsgn\ z) = hRe(z)/hcm\ od\ z$

⟨proof⟩

**lemma** *hIm-hsgn* [simp]:  $!!z. \text{hIm}(\text{hsgn } z) = \text{hIm}(z)/\text{hcmod } z$   
 ⟨proof⟩

**lemma** *HComplex-inverse*:  
 $!!x \ y. \text{inverse } (\text{HComplex } x \ y) =$   
 $\text{HComplex } (x/(x^2 + y^2)) \ (-y/(x^2 + y^2))$   
 ⟨proof⟩

**lemma** *hRe-mult-i-eq*[simp]:  
 $!!y. \text{hRe } (iii * \text{hcomplex-of-hypreal } y) = 0$   
 ⟨proof⟩

**lemma** *hIm-mult-i-eq* [simp]:  
 $!!y. \text{hIm } (iii * \text{hcomplex-of-hypreal } y) = y$   
 ⟨proof⟩

**lemma** *hcmmod-mult-i* [simp]:  $!!y. \text{hcmmod } (iii * \text{hcomplex-of-hypreal } y) = \text{abs } y$   
 ⟨proof⟩

**lemma** *hcmmod-mult-i2* [simp]:  $!!y. \text{hcmmod } (\text{hcomplex-of-hypreal } y * iii) = \text{abs } y$   
 ⟨proof⟩

**lemma** *cos-harg-i-mult-zero* [simp]:  
 $!!y. y \neq 0 \implies (*f* \text{cos}) (\text{harg}(\text{HComplex } 0 \ y)) = 0$   
 ⟨proof⟩

**lemma** *hcomplex-of-hypreal-zero-iff* [simp]:  
 $!!y. (\text{hcomplex-of-hypreal } y = 0) = (y = 0)$   
 ⟨proof⟩

### 9.13 Polar Form for Nonstandard Complex Numbers

**lemma** *complex-split-polar2*:  
 $\forall n. \exists r \ a. (z \ n) = \text{complex-of-real } r * (\text{Complex } (\text{cos } a) \ (\text{sin } a))$   
 ⟨proof⟩

**lemma** *hcomplex-split-polar*:  
 $!!z. \exists r \ a. z = \text{hcomplex-of-hypreal } r * (\text{HComplex}(( *f* \text{cos}) \ a)(( *f* \text{sin}) \ a))$   
 ⟨proof⟩

**lemma** *hcis-eq*:  
 $!!a. \text{hcis } a =$   
 $(\text{hcomplex-of-hypreal}(( *f* \text{cos}) \ a) +$

*iii* \* *hcomplex-of-hypreal*(( \*f\* sin) a))  
 ⟨proof⟩

**lemma** *hrcis-Ex*: !!z. ∃ r a. z = *hrcis* r a  
 ⟨proof⟩

**lemma** *hRe-hcomplex-polar* [simp]:  
 !!r a. *hRe* (*hcomplex-of-hypreal* r \* *HComplex* (( \*f\* cos) a) (( \*f\* sin) a)) =  
 r \* ( \*f\* cos) a  
 ⟨proof⟩

**lemma** *hRe-hrcis* [simp]: !!r a. *hRe*(*hrcis* r a) = r \* ( \*f\* cos) a  
 ⟨proof⟩

**lemma** *hIm-hcomplex-polar* [simp]:  
 !!r a. *hIm* (*hcomplex-of-hypreal* r \* *HComplex* (( \*f\* cos) a) (( \*f\* sin) a)) =  
 r \* ( \*f\* sin) a  
 ⟨proof⟩

**lemma** *hIm-hrcis* [simp]: !!r a. *hIm*(*hrcis* r a) = r \* ( \*f\* sin) a  
 ⟨proof⟩

**lemma** *hcmmod-unit-one* [simp]:  
 !!a. *hcmmod* (*HComplex* (( \*f\* cos) a) (( \*f\* sin) a)) = 1  
 ⟨proof⟩

**lemma** *hcmmod-complex-polar* [simp]:  
 !!r a. *hcmmod* (*hcomplex-of-hypreal* r \* *HComplex* (( \*f\* cos) a) (( \*f\* sin) a)) =  
 abs r  
 ⟨proof⟩

**lemma** *hcmmod-hrcis* [simp]: !!r a. *hcmmod*(*hrcis* r a) = abs r  
 ⟨proof⟩

**lemma** *hcis-hrcis-eq*: !!a. *hcis* a = *hrcis* 1 a  
 ⟨proof⟩

**declare** *hcis-hrcis-eq* [symmetric, simp]

**lemma** *hrcis-mult*:  
 !!a b r1 r2. *hrcis* r1 a \* *hrcis* r2 b = *hrcis* (r1\*r2) (a + b)  
 ⟨proof⟩

**lemma** *hcis-mult*: !!a b. *hcis* a \* *hcis* b = *hcis* (a + b)  
 ⟨proof⟩

**lemma** *hcis-zero* [*simp*]:  $hcis\ 0 = 1$   
 ⟨*proof*⟩

**lemma** *hrcis-zero-mod* [*simp*]:  $!!a. hrcis\ 0\ a = 0$   
 ⟨*proof*⟩

**lemma** *hrcis-zero-arg* [*simp*]:  $!!r. hrcis\ r\ 0 = hcomplex-of-hypreal\ r$   
 ⟨*proof*⟩

**lemma** *hcomplex-i-mult-minus* [*simp*]:  $!!x. iii * (iii * x) = - x$   
 ⟨*proof*⟩

**lemma** *hcomplex-i-mult-minus2* [*simp*]:  $iii * iii * x = - x$   
 ⟨*proof*⟩

**lemma** *hcis-hypreal-of-nat-Suc-mult*:  
 $!!a. hcis\ (hypreal-of-nat\ (Suc\ n) * a) =$   
 $hcis\ a * hcis\ (hypreal-of-nat\ n * a)$   
 ⟨*proof*⟩

**lemma** *NSDeMoivre*:  $!!a. (hcis\ a) ^ n = hcis\ (hypreal-of-nat\ n * a)$   
 ⟨*proof*⟩

**lemma** *hcis-hypreal-of-hypnat-Suc-mult*:  
 $!! a\ n. hcis\ (hypreal-of-hypnat\ (n + 1) * a) =$   
 $hcis\ a * hcis\ (hypreal-of-hypnat\ n * a)$   
 ⟨*proof*⟩

**lemma** *NSDeMoivre-ext*:  
 $!!a\ n. (hcis\ a) pow\ n = hcis\ (hypreal-of-hypnat\ n * a)$   
 ⟨*proof*⟩

**lemma** *NSDeMoivre2*:  
 $!!a\ r. (hrcis\ r\ a) ^ n = hrcis\ (r ^ n) (hypreal-of-nat\ n * a)$   
 ⟨*proof*⟩

**lemma** *DeMoivre2-ext*:  
 $!! a\ r\ n. (hrcis\ r\ a) pow\ n = hrcis\ (r pow\ n) (hypreal-of-hypnat\ n * a)$   
 ⟨*proof*⟩

**lemma** *hcis-inverse* [*simp*]:  $!!a. inverse(hcis\ a) = hcis\ (-a)$   
 ⟨*proof*⟩

**lemma** *hrcis-inverse*:  $!!a\ r. inverse(hrcis\ r\ a) = hrcis\ (inverse\ r) (-a)$   
 ⟨*proof*⟩

**lemma** *hRe-hcis* [*simp*]:  $!!a. hRe(hcis\ a) = (*f* cos) a$   
 ⟨*proof*⟩

**lemma** *hIm-hcis [simp]: !!a. hIm(hcis a) = (\*f\* sin) a*  
 ⟨proof⟩

**lemma** *cos-n-hRe-hcis-pow-n: (\*f\* cos) (hypreal-of-nat n \* a) = hRe(hcis a ^ n)*  
 ⟨proof⟩

**lemma** *sin-n-hIm-hcis-pow-n: (\*f\* sin) (hypreal-of-nat n \* a) = hIm(hcis a ^ n)*  
 ⟨proof⟩

**lemma** *cos-n-hRe-hcis-hcpow-n: (\*f\* cos) (hypreal-of-hypnat n \* a) = hRe(hcis a pow n)*  
 ⟨proof⟩

**lemma** *sin-n-hIm-hcis-hcpow-n: (\*f\* sin) (hypreal-of-hypnat n \* a) = hIm(hcis a pow n)*  
 ⟨proof⟩

**lemma** *hexpi-add: !!a b. hexpi(a + b) = hexpi(a) \* hexpi(b)*  
 ⟨proof⟩

### 9.14 *hcomplex-of-complex: the Injection from type complex to hcomplex*

**lemma** *inj-hcomplex-of-complex: inj(hcomplex-of-complex)*  
 ⟨proof⟩

**lemma** *hcomplex-of-complex-i: iii = hcomplex-of-complex ii*  
 ⟨proof⟩

**lemma** *hRe-hcomplex-of-complex:*  
*hRe (hcomplex-of-complex z) = hypreal-of-real (Re z)*  
 ⟨proof⟩

**lemma** *hIm-hcomplex-of-complex:*  
*hIm (hcomplex-of-complex z) = hypreal-of-real (Im z)*  
 ⟨proof⟩

**lemma** *hcmmod-hcomplex-of-complex:*  
*hcmmod (hcomplex-of-complex x) = hypreal-of-real (cmmod x)*  
 ⟨proof⟩

### 9.15 Numerals and Arithmetic

**lemma** *hcomplex-of-hypreal-eq-hcomplex-of-complex:*  
*hcomplex-of-hypreal (hypreal-of-real x) =*  
*hcomplex-of-complex (complex-of-real x)*  
 ⟨proof⟩

**lemma** *hcomplex-hypreal-numeral*:

$hcomplex\text{-of-complex}(\text{numeral } w) = hcomplex\text{-of-hypreal}(\text{numeral } w)$   
 $\langle proof \rangle$

**lemma** *hcomplex-hypreal-neg-numeral*:

$hcomplex\text{-of-complex}(\text{neg-numeral } w) = hcomplex\text{-of-hypreal}(\text{neg-numeral } w)$   
 $\langle proof \rangle$

**lemma** *hcomplex-numeral-hcnj* [simp]:

$hcnj(\text{numeral } v :: hcomplex) = \text{numeral } v$   
 $\langle proof \rangle$

**lemma** *hcomplex-numeral-hcmod* [simp]:

$hcmod(\text{numeral } v :: hcomplex) = (\text{numeral } v :: hypreal)$   
 $\langle proof \rangle$

**lemma** *hcomplex-neg-numeral-hcmod* [simp]:

$hcmod(\text{neg-numeral } v :: hcomplex) = (\text{numeral } v :: hypreal)$   
 $\langle proof \rangle$

**lemma** *hcomplex-numeral-hRe* [simp]:

$hRe(\text{numeral } v :: hcomplex) = \text{numeral } v$   
 $\langle proof \rangle$

**lemma** *hcomplex-numeral-hIm* [simp]:

$hIm(\text{numeral } v :: hcomplex) = 0$   
 $\langle proof \rangle$

end

## 10 Star: Star-Transforms in Non-Standard Analysis

**theory** *Star*  
**imports** *NSA*  
**begin**

**definition**

$starset\text{-}n :: (\text{nat} \Rightarrow 'a \text{ set}) \Rightarrow 'a \text{ star set } (*sn* - [80] 80)$  **where**  
 $*sn* \text{ } As = Iset (star\text{-}n \text{ } As)$

**definition**

$InternalSets :: 'a \text{ star set set}$  **where**  
 $InternalSets = \{X. \exists As. X = *sn* \text{ } As\}$

**definition**

*is-starext* :: [*'a star => 'a star, 'a => 'a*] => *bool* **where**  
*is-starext* *F f* = ( $\forall x y. \exists X \in \text{Rep-star}(x). \exists Y \in \text{Rep-star}(y).$   
 $((y = (F x)) = (\{n. Y n = f(X n)\} : \text{FreeUltrafilterNat}))$ )

**definition**

*starfun-n* :: (*nat => ('a => 'b)*) => *'a star => 'b star* (*\*fn\* - [80] 80*) **where**  
*\*fn\* F* = *Ifun (star-n F)*

**definition**

*InternalFuns* :: (*'a star => 'b star*) *set* **where**  
*InternalFuns* = {*X. \exists F. X = \*fn\* F*}

**lemma** *no-choice*:  $\forall x. \exists y. Q x y \implies \exists (f :: 'a \Rightarrow \text{nat}). \forall x. Q x (f x)$   
 $\langle \text{proof} \rangle$

## 10.1 Properties of the Star-transform Applied to Sets of Reals

**lemma** *STAR-star-of-image-subset*: *star-of ' A <= \*s\* A*  
 $\langle \text{proof} \rangle$

**lemma** *STAR-hypreal-of-real-Int*: *\*s\* X Int Reals = hypreal-of-real ' X*  
 $\langle \text{proof} \rangle$

**lemma** *STAR-star-of-Int*: *\*s\* X Int Standard = star-of ' X*  
 $\langle \text{proof} \rangle$

**lemma** *lemma-not-hyprealA*:  $x \notin \text{hypreal-of-real ' A} \implies \forall y \in A. x \neq \text{hypreal-of-real } y$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-not-starA*:  $x \notin \text{star-of ' A} \implies \forall y \in A. x \neq \text{star-of } y$   
 $\langle \text{proof} \rangle$

**lemma** *lemma-Compl-eq*:  $-\{n. X n = xa\} = \{n. X n \neq xa\}$   
 $\langle \text{proof} \rangle$

**lemma** *STAR-real-seq-to-hypreal*:  
 $\forall n. (X n) \notin M \implies \text{star-n } X \notin *s* M$   
 $\langle \text{proof} \rangle$

**lemma** *STAR-singleton*: *\*s\* {x} = {star-of x}*

*<proof>*

**lemma** *STAR-not-mem*:  $x \notin F \implies \text{star-of } x \notin \text{*s* } F$

*<proof>*

**lemma** *STAR-subset-closed*:  $[| x : \text{*s* } A; A \leq B |] \implies x : \text{*s* } B$

*<proof>*

Nonstandard extension of a set (defined using a constant sequence) as a special case of an internal set

**lemma** *starset-n-starset*:  $\forall n. (As\ n = A) \implies \text{*sn* } As = \text{*s* } A$

*<proof>*

**lemma** *starfun-n-starfun*:  $\forall n. (F\ n = f) \implies \text{*fn* } F = \text{*f* } f$

*<proof>*

**lemma** *hrabs-is-starext-rabs*: *is-starext abs abs*

*<proof>*

Nonstandard extension of functions

**lemma** *starfun*:

$(\text{*f* } f) (\text{star-n } X) = \text{star-n } (\%n. f (X\ n))$

*<proof>*

**lemma** *starfun-if-eq*:

$!!w. w \neq \text{star-of } x$

$\implies (\text{*f* } (\lambda z. \text{if } z = x \text{ then } a \text{ else } g\ z))\ w = (\text{*f* } g)\ w$

*<proof>*

**lemma** *starfun-mult*:  $!!x. (\text{*f* } f)\ x * (\text{*f* } g)\ x = (\text{*f* } (\%x. f\ x * g\ x))\ x$

*<proof>*

**declare** *starfun-mult* [*symmetric, simp*]

**lemma** *starfun-add*:  $!!x. (\text{*f* } f)\ x + (\text{*f* } g)\ x = (\text{*f* } (\%x. f\ x + g\ x))\ x$

$\langle proof \rangle$   
**declare** *starfun-add* [*symmetric, simp*]

**lemma** *starfun-minus*:  $!!x. - (*f* f) x = (*f* (\%x. - f x)) x$   
 $\langle proof \rangle$   
**declare** *starfun-minus* [*symmetric, simp*]

**lemma** *starfun-add-minus*:  $!!x. (*f* f) x + -(*f* g) x = (*f* (\%x. f x + -g x)) x$   
 $\langle proof \rangle$   
**declare** *starfun-add-minus* [*symmetric, simp*]

**lemma** *starfun-diff*:  $!!x. (*f* f) x - (*f* g) x = (*f* (\%x. f x - g x)) x$   
 $\langle proof \rangle$   
**declare** *starfun-diff* [*symmetric, simp*]

**lemma** *starfun-o2*:  $(\%x. (*f* f) (( *f* g) x)) = *f* (\%x. f (g x))$   
 $\langle proof \rangle$

**lemma** *starfun-o*:  $(*f* f) o (*f* g) = (*f* (f o g))$   
 $\langle proof \rangle$

NS extension of constant function

**lemma** *starfun-const-fun* [*simp*]:  $!!x. (*f* (\%x. k)) x = star-of k$   
 $\langle proof \rangle$

the NS extension of the identity function

**lemma** *starfun-Id* [*simp*]:  $!!x. (*f* (\%x. x)) x = x$   
 $\langle proof \rangle$

**lemma** *starfun-Idfun-approx*:  
 $x @= star-of a ==> (*f* (\%x. x)) x @= star-of a$   
 $\langle proof \rangle$

The Star-function is a (nonstandard) extension of the function

**lemma** *is-starext-starfun*:  $is-starext (*f* f) f$   
 $\langle proof \rangle$

Any nonstandard extension is in fact the Star-function

**lemma** *is-starfun-starext*:  $is-starext F f ==> F = *f* f$   
 $\langle proof \rangle$

**lemma** *is-starext-starfun-iff*:  $(is-starext F f) = (F = *f* f)$

*<proof>*

extended function has same solution as its standard version for real arguments. i.e they are the same for all real arguments

**lemma** *starfun-eq*: ( $*f*$   $f$ ) (*star-of*  $a$ ) = *star-of* ( $f$   $a$ )

*<proof>*

**lemma** *starfun-approx*: ( $*f*$   $f$ ) (*star-of*  $a$ ) @= *star-of* ( $f$   $a$ )

*<proof>*

**lemma** *starfun-lambda-cancel*:

!! $x'$ . ( $*f*$  ( $\%h$ .  $f$  ( $x + h$ )))  $x'$  = ( $*f*$   $f$ ) (*star-of*  $x + x'$ )

*<proof>*

**lemma** *starfun-lambda-cancel2*:

( $*f*$  ( $\%h$ .  $f(g(x + h))$ ))  $x'$  = ( $*f*$  ( $f$   $o$   $g$ )) (*star-of*  $x + x'$ )

*<proof>*

**lemma** *starfun-mult-HFinite-approx*:

**fixes**  $l$   $m$  :: ' $a$ ::*real-normed-algebra* *star*

**shows** [| ( $*f*$   $f$ )  $x$  @=  $l$ ; ( $*f*$   $g$ )  $x$  @=  $m$ ;

$l$ : *HFinite*;  $m$ : *HFinite*

|| ==> ( $*f*$  ( $\%x$ .  $f$   $x$  \*  $g$   $x$ ))  $x$  @=  $l$  \*  $m$

*<proof>*

**lemma** *starfun-add-approx*: [| ( $*f*$   $f$ )  $x$  @=  $l$ ; ( $*f*$   $g$ )  $x$  @=  $m$

|| ==> ( $*f*$  ( $\%x$ .  $f$   $x$  +  $g$   $x$ ))  $x$  @=  $l$  +  $m$

*<proof>*

Examples: *hrabs* is nonstandard extension of *rabs* inverse is nonstandard extension of inverse

**lemma** *starfun-rabs-hrabs*:  $*f*$  *abs* = *abs*

*<proof>*

**lemma** *starfun-inverse-inverse* [*simp*]: ( $*f*$  *inverse*)  $x$  = *inverse*( $x$ )

*<proof>*

**lemma** *starfun-inverse*: !! $x$ . *inverse* (( $*f*$   $f$ )  $x$ ) = ( $*f*$  ( $\%x$ . *inverse* ( $f$   $x$ )))  $x$

*<proof>*

**declare** *starfun-inverse* [*symmetric*, *simp*]

**lemma** *starfun-divide*: !! $x$ . ( $*f*$   $f$ )  $x$  / ( $*f*$   $g$ )  $x$  = ( $*f*$  ( $\%x$ .  $f$   $x$  /  $g$   $x$ ))  $x$

*<proof>*

**declare** *starfun-divide* [*symmetric*, *simp*]

**lemma** *starfun-inverse2*: !! $x$ . *inverse* (( $*f*$   $f$ )  $x$ ) = ( $*f*$  ( $\%x$ . *inverse* ( $f$   $x$ )))  $x$

*<proof>*

General lemma/theorem needed for proofs in elementary topology of the reals

**lemma** *starfun-mem-starset*:

$$\begin{aligned} &!!x. ( *f* f ) x : *s* A ==> x : *s* \{x. f x \in A\} \\ &\langle proof \rangle \end{aligned}$$

Alternative definition for hrabs with rabs function applied entrywise to equivalence class representative. This is easily proved using starfun and ns extension thm

**lemma** *hypreal-hrabs*:

$$\begin{aligned} &abs (star-n X) = star-n (\%n. abs (X n)) \\ &\langle proof \rangle \end{aligned}$$

nonstandard extension of set through nonstandard extension of rabs function i.e hrabs. A more general result should be where we replace rabs by some arbitrary function f and hrabs by its NS extension. See second NS set extension below.

**lemma** *STAR-rabs-add-minus*:

$$\begin{aligned} &*s* \{x. abs (x + - y) < r\} = \\ &\{x. abs(x + -star-of y) < star-of r\} \\ &\langle proof \rangle \end{aligned}$$

**lemma** *STAR-starfun-rabs-add-minus*:

$$\begin{aligned} &*s* \{x. abs (f x + - y) < r\} = \\ &\{x. abs(( *f* f ) x + -star-of y) < star-of r\} \\ &\langle proof \rangle \end{aligned}$$

Another characterization of Infinitesimal and one of @= relation. In this theory since *hypreal-hrabs* proved here. Maybe move both theorems??

**lemma** *Infinitesimal-FreeUltrafilterNat-iff2*:

$$\begin{aligned} &(star-n X \in Infinitesimal) = \\ &(\forall m. \{n. norm(X n) < inverse(real(Suc m))\} \\ &\quad \in FreeUltrafilterNat) \\ &\langle proof \rangle \end{aligned}$$

**lemma** *HNatInfinite-inverse-Infinitesimal [simp]*:

$$\begin{aligned} &n \in HNatInfinite ==> inverse (hypreal-of-hypnat n) \in Infinitesimal \\ &\langle proof \rangle \end{aligned}$$

**lemma** *approx-FreeUltrafilterNat-iff*:  $star-n X @= star-n Y =$

$$\begin{aligned} &(\forall r>0. \{n. norm (X n - Y n) < r\} : FreeUltrafilterNat) \\ &\langle proof \rangle \end{aligned}$$

**lemma** *approx-FreeUltrafilterNat-iff2*:  $star-n X @= star-n Y =$

$$\begin{aligned} &(\forall m. \{n. norm (X n - Y n) < \\ &\quad inverse(real(Suc m))\} : FreeUltrafilterNat) \\ &\langle proof \rangle \end{aligned}$$

**lemma** *inj-starfun*: *inj starfun*

*<proof>*

**end**

## 11 NatStar: Star-transforms for the Hypernaturals

**theory** *NatStar*

**imports** *Star*

**begin**

**lemma** *star-n-eq-starfun-whn*:  $star\ n\ X = (*f*\ X)\ whn$

*<proof>*

**lemma** *starset-n-Un*:  $*sn*\ (\%n.\ (A\ n)\ Un\ (B\ n)) = *sn*\ A\ Un\ *sn*\ B$

*<proof>*

**lemma** *InternalSets-Un*:

$[[\ X \in InternalSets; Y \in InternalSets\ ]]$

$==>\ (X\ Un\ Y) \in InternalSets$

*<proof>*

**lemma** *starset-n-Int*:

$*sn*\ (\%n.\ (A\ n)\ Int\ (B\ n)) = *sn*\ A\ Int\ *sn*\ B$

*<proof>*

**lemma** *InternalSets-Int*:

$[[\ X \in InternalSets; Y \in InternalSets\ ]]$

$==>\ (X\ Int\ Y) \in InternalSets$

*<proof>*

**lemma** *starset-n-Compl*:  $*sn*\ ((\%n.\ -\ A\ n)) = -(*sn*\ A)$

*<proof>*

**lemma** *InternalSets-Compl*:  $X \in InternalSets ==>\ -X \in InternalSets$

*<proof>*

**lemma** *starset-n-diff*:  $*sn*\ (\%n.\ (A\ n) - (B\ n)) = *sn*\ A - *sn*\ B$

*<proof>*

**lemma** *InternalSets-diff*:

$[[\ X \in InternalSets; Y \in InternalSets\ ]]$

$==>\ (X - Y) \in InternalSets$

*<proof>*

**lemma** *NatStar-SHNat-subset*:  $Nats \leq ** (UNIV:: nat\ set)$   
 $\langle proof \rangle$

**lemma** *NatStar-hypreal-of-real-Int*:  
 $** X\ Int\ Nats = hypnat-of-nat\ ' X$   
 $\langle proof \rangle$

**lemma** *starset-starset-n-eq*:  $** X = *sn* (\%n. X)$   
 $\langle proof \rangle$

**lemma** *InternalSets-starset-n [simp]*:  $(** X) \in InternalSets$   
 $\langle proof \rangle$

**lemma** *InternalSets-UNIV-diff*:  
 $X \in InternalSets ==> UNIV - X \in InternalSets$   
 $\langle proof \rangle$

## 11.1 Nonstandard Extensions of Functions

Example of transfer of a property from reals to hyperreals — used for limit comparison of sequences

**lemma** *starfun-le-mono*:  
 $\forall n. N \leq n \longrightarrow f\ n \leq g\ n$   
 $==> \forall n. hypnat-of-nat\ N \leq n \longrightarrow (** f)\ n \leq (** g)\ n$   
 $\langle proof \rangle$

**lemma** *starfun-less-mono*:  
 $\forall n. N \leq n \longrightarrow f\ n < g\ n$   
 $==> \forall n. hypnat-of-nat\ N \leq n \longrightarrow (** f)\ n < (** g)\ n$   
 $\langle proof \rangle$

Nonstandard extension when we increment the argument by one

**lemma** *starfun-shift-one*:  
 $!!N. (** (\%n. f\ (Suc\ n)))\ N = (** f)\ (N + (1::hypnat))$   
 $\langle proof \rangle$

Nonstandard extension with absolute value

**lemma** *starfun-abs*:  $!!N. (** (\%n. abs\ (f\ n)))\ N = abs\ (** f)\ N$   
 $\langle proof \rangle$

The hyperpow function as a nonstandard extension of realpow

**lemma** *starfun-pow*:  $!!N. (** (\%n. r\ ^\ n))\ N = (hypreal-of-real\ r)\ pow\ N$   
 $\langle proof \rangle$

**lemma** *starfun-pow2*:  
 $!!N. (** (\%n. (X\ n)\ ^\ m))\ N = (** X)\ N\ pow\ hypnat-of-nat\ m$   
 $\langle proof \rangle$

**lemma** *starfun-pow3*:  $!!R. (*f* (\%r. r \wedge n)) R = (R) \text{ pow hypnat-of-nat } n$   
 ⟨proof⟩

The *hypreal-of-hypnat* function as a nonstandard extension of *real-of-nat*

**lemma** *starfunNat-real-of-nat*:  $(*f* \text{ real}) = \text{hypreal-of-hypnat}$   
 ⟨proof⟩

**lemma** *starfun-inverse-real-of-nat-eq*:

$N \in \text{HNatInfinite}$   
 $\implies (*f* (\%x::\text{nat. inverse}(\text{real } x))) N = \text{inverse}(\text{hypreal-of-hypnat } N)$   
 ⟨proof⟩

Internal functions - some redundancy with *\*f\** now

**lemma** *starfun-n*:  $(*fn* f) (\text{star-n } X) = \text{star-n } (\%n. f \ n \ (X \ n))$   
 ⟨proof⟩

Multiplication:  $(*fn) \ x \ (*gn) = *(fn \ x \ gn)$

**lemma** *starfun-n-mult*:

$(*fn* f) \ z \ (*fn* g) \ z = (*fn* (\%i \ x. f \ i \ x \ * \ g \ i \ x)) \ z$   
 ⟨proof⟩

Addition:  $(*fn) + (*gn) = *(fn + gn)$

**lemma** *starfun-n-add*:

$(*fn* f) \ z + (*fn* g) \ z = (*fn* (\%i \ x. f \ i \ x + g \ i \ x)) \ z$   
 ⟨proof⟩

Subtraction:  $(*fn) - (*gn) = *(fn + - \ gn)$

**lemma** *starfun-n-add-minus*:

$(*fn* f) \ z + -( *fn* g) \ z = (*fn* (\%i \ x. f \ i \ x + -g \ i \ x)) \ z$   
 ⟨proof⟩

Composition:  $(*fn) \ o \ (*gn) = *(fn \ o \ gn)$

**lemma** *starfun-n-const-fun* [simp]:

$(*fn* (\%i \ x. k)) \ z = \text{star-of } k$   
 ⟨proof⟩

**lemma** *starfun-n-minus*:  $-( *fn* f) \ x = (*fn* (\%i \ x. - (f \ i) \ x)) \ x$   
 ⟨proof⟩

**lemma** *starfun-n-eq* [simp]:

$(*fn* f) (\text{star-of } n) = \text{star-n } (\%i. f \ i \ n)$   
 ⟨proof⟩

**lemma** *starfun-eq-iff*:  $(( *f* f) = (*f* g)) = (f = g)$

⟨proof⟩

**lemma** *starfunNat-inverse-real-of-nat-Infinitesimal* [simp]:

$N \in \text{HNatInfinite} \implies (*f* (\%x. \text{inverse} (\text{real } x))) N \in \text{Infinitesimal}$   
 <proof>

## 11.2 Nonstandard Characterization of Induction

**lemma** *hypnat-induct-obj*:

!!n. (( \*p\* P) (0::hypnat) &  
 (∀ n. ( \*p\* P)(n) --> ( \*p\* P)(n + 1)))  
 --> ( \*p\* P)(n)  
 <proof>

**lemma** *hypnat-induct*:

!!n. [| ( \*p\* P) (0::hypnat);  
 !!n. ( \*p\* P)(n) ==> ( \*p\* P)(n + 1)|]  
 ==> ( \*p\* P)(n)  
 <proof>

**lemma** *starP2-eq-iff*: ( \*p2\* (op =)) = (op =)  
 <proof>

**lemma** *starP2-eq-iff2*: ( \*p2\* (%x y. x = y)) X Y = (X = Y)  
 <proof>

**lemma** *nonempty-nat-set-Least-mem*:

c ∈ (S :: nat set) ==> (LEAST n. n ∈ S) ∈ S  
 <proof>

**lemma** *nonempty-set-star-has-least*:

!!S::nat set star. Iset S ≠ {} ==> ∃ n ∈ Iset S. ∀ m ∈ Iset S. n ≤ m  
 <proof>

**lemma** *nonempty-InternalNatSet-has-least*:

[| (S::hypnat set) ∈ InternalSets; S ≠ {} |] ==> ∃ n ∈ S. ∀ m ∈ S. n ≤ m  
 <proof>

Goldblatt page 129 Thm 11.3.2

**lemma** *internal-induct-lemma*:

!!X::nat set star. [| (0::hypnat) ∈ Iset X; ∀ n. n ∈ Iset X --> n + 1 ∈ Iset X |]  
 ==> Iset X = (UNIV:: hypnat set)  
 <proof>

**lemma** *internal-induct*:

[| X ∈ InternalSets; (0::hypnat) ∈ X; ∀ n. n ∈ X --> n + 1 ∈ X |]  
 ==> X = (UNIV:: hypnat set)  
 <proof>

**end**

## 12 HSEQ: Sequences and Convergence (Nonstandard)

**theory** *HSEQ*  
**imports** *SEQ NatStar*  
**begin**

**definition**

*NSLIMSEQ* ::  $[nat \Rightarrow 'a::real-normed-vector, 'a] \Rightarrow bool$   
 $(((-)/ \text{-----}NS> (-)) [60, 60] 60)$  **where**  
 — Nonstandard definition of convergence of sequence  
 $X \text{-----}NS> L = (\forall N \in HNatInfinite. (*f* X) N \approx star-of L)$

**definition**

*nslim* ::  $(nat \Rightarrow 'a::real-normed-vector) \Rightarrow 'a$  **where**  
 — Nonstandard definition of limit using choice operator  
 $nslim X = (THE L. X \text{-----}NS> L)$

**definition**

*NSconvergent* ::  $(nat \Rightarrow 'a::real-normed-vector) \Rightarrow bool$  **where**  
 — Nonstandard definition of convergence  
 $NSconvergent X = (\exists L. X \text{-----}NS> L)$

**definition**

*NSBseq* ::  $(nat \Rightarrow 'a::real-normed-vector) \Rightarrow bool$  **where**  
 — Nonstandard definition for bounded sequence  
 $NSBseq X = (\forall N \in HNatInfinite. (*f* X) N : HFinite)$

**definition**

*NSCauchy* ::  $(nat \Rightarrow 'a::real-normed-vector) \Rightarrow bool$  **where**  
 — Nonstandard definition  
 $NSCauchy X = (\forall M \in HNatInfinite. \forall N \in HNatInfinite. (*f* X) M \approx (*f* X) N)$

### 12.1 Limits of Sequences

**lemma** *NSLIMSEQ-iff*:

$(X \text{-----}NS> L) = (\forall N \in HNatInfinite. (*f* X) N \approx star-of L)$   
 $\langle proof \rangle$

**lemma** *NSLIMSEQ-I*:

$(\bigwedge N. N \in HNatInfinite \implies starfun X N \approx star-of L) \implies X \text{-----}NS> L$   
 $\langle proof \rangle$

**lemma** *NSLIMSEQ-D*:

$\llbracket X \text{-----}NS> L; N \in HNatInfinite \rrbracket \implies starfun X N \approx star-of L$

*<proof>*

**lemma** *NSLIMSEQ-const*:  $(\%n. k) \text{ ----NS} > k$   
*<proof>*

**lemma** *NSLIMSEQ-add*:

$[[ X \text{ ----NS} > a; Y \text{ ----NS} > b ]] \implies (\%n. X n + Y n) \text{ ----NS} > a + b$   
*<proof>*

**lemma** *NSLIMSEQ-add-const*:  $f \text{ ----NS} > a \implies (\%n.(f n + b)) \text{ ----NS} > a + b$   
*<proof>*

**lemma** *NSLIMSEQ-mult*:

**fixes**  $a b :: 'a::\text{real-normed-algebra}$   
**shows**  $[[ X \text{ ----NS} > a; Y \text{ ----NS} > b ]] \implies (\%n. X n * Y n) \text{ ----NS} > a * b$   
*<proof>*

**lemma** *NSLIMSEQ-minus*:  $X \text{ ----NS} > a \implies (\%n. -(X n)) \text{ ----NS} > -a$   
*<proof>*

**lemma** *NSLIMSEQ-minus-cancel*:  $(\%n. -(X n)) \text{ ----NS} > -a \implies X \text{ ----NS} > a$   
*<proof>*

**lemma** *NSLIMSEQ-add-minus*:

$[[ X \text{ ----NS} > a; Y \text{ ----NS} > b ]] \implies (\%n. X n + -Y n) \text{ ----NS} > a + -b$   
*<proof>*

**lemma** *NSLIMSEQ-diff*:

$[[ X \text{ ----NS} > a; Y \text{ ----NS} > b ]] \implies (\%n. X n - Y n) \text{ ----NS} > a - b$   
*<proof>*

**lemma** *NSLIMSEQ-diff-const*:  $f \text{ ----NS} > a \implies (\%n.(f n - b)) \text{ ----NS} > a - b$   
*<proof>*

**lemma** *NSLIMSEQ-inverse*:

**fixes**  $a :: 'a::\text{real-normed-div-algebra}$   
**shows**  $[[ X \text{ ----NS} > a; a \sim 0 ]] \implies (\%n. \text{inverse}(X n)) \text{ ----NS} > \text{inverse}(a)$   
*<proof>*

**lemma** *NSLIMSEQ-mult-inverse*:

**fixes**  $a\ b :: 'a::\text{real-normed-field}$   
**shows**  
 $\llbracket X \text{ ----NS} > a; Y \text{ ----NS} > b; b \sim 0 \rrbracket \implies (\%n. X\ n / Y\ n) \text{ ----NS} > a/b$   
 $\langle\text{proof}\rangle$

**lemma** *starfun-hnorm*:  $\bigwedge x. \text{hnorm} (( *f* f) x) = ( *f* (\lambda x. \text{norm} (f x))) x$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-norm*:  $X \text{ ----NS} > a \implies (\lambda n. \text{norm} (X\ n)) \text{ ----NS} > \text{norm } a$   
 $\langle\text{proof}\rangle$

Uniqueness of limit

**lemma** *NSLIMSEQ-unique*:  $\llbracket X \text{ ----NS} > a; X \text{ ----NS} > b \rrbracket \implies a = b$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-pow* [rule-format]:  
**fixes**  $a :: 'a::\{\text{real-normed-algebra,power}\}$   
**shows**  $(X \text{ ----NS} > a) \text{ ---} > ((\%n. (X\ n) ^ m) \text{ ----NS} > a ^ m)$   
 $\langle\text{proof}\rangle$

We can now try and derive a few properties of sequences, starting with the limit comparison property for sequences.

**lemma** *NSLIMSEQ-le*:  
 $\llbracket f \text{ ----NS} > l; g \text{ ----NS} > m; \exists N. \forall n \geq N. f(n) \leq g(n) \rrbracket \implies l \leq (m::\text{real})$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-le-const*:  $\llbracket X \text{ ----NS} > (r::\text{real}); \forall n. a \leq X\ n \rrbracket \implies a \leq r$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-le-const2*:  $\llbracket X \text{ ----NS} > (r::\text{real}); \forall n. X\ n \leq a \rrbracket \implies r \leq a$   
 $\langle\text{proof}\rangle$

Shift a convergent series by 1: By the equivalence between Cauchiness and convergence and because the successor of an infinite hypernatural is also infinite.

**lemma** *NSLIMSEQ-Suc*:  $f \text{ ----NS} > l \implies (\%n. f(\text{Suc } n)) \text{ ----NS} > l$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-imp-Suc*:  $(\%n. f(\text{Suc } n)) \text{ ----NS} > l \implies f \text{ ----NS} > l$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIMSEQ-Suc-iff*:  $((\%n. f(\text{Suc } n)) \text{----} \text{NS} > l) = (f \text{----} \text{NS} > l)$   
 ⟨proof⟩

### 12.1.1 Equivalence of LIMSEQ and NSLIMSEQ

**lemma** *LIMSEQ-NSLIMSEQ*:

**assumes**  $X: X \text{----} > L$  **shows**  $X \text{----} \text{NS} > L$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-LIMSEQ*:

**assumes**  $X: X \text{----} \text{NS} > L$  **shows**  $X \text{----} > L$   
 ⟨proof⟩

**theorem** *LIMSEQ-NSLIMSEQ-iff*:  $(f \text{----} > L) = (f \text{----} \text{NS} > L)$   
 ⟨proof⟩

### 12.1.2 Derived theorems about NSLIMSEQ

We prove the NS version from the standard one, since the NS proof seems more complicated than the standard one above!

**lemma** *NSLIMSEQ-norm-zero*:  $((\lambda n. \text{norm } (X n)) \text{----} \text{NS} > 0) = (X \text{----} \text{NS} > 0)$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-rabs-zero*:  $((\%n. |f n|) \text{----} \text{NS} > 0) = (f \text{----} \text{NS} > (0::\text{real}))$   
 ⟨proof⟩

Generalization to other limits

**lemma** *NSLIMSEQ-imp-rabs*:  $f \text{----} \text{NS} > (l::\text{real}) \implies (\%n. |f n|) \text{----} \text{NS} > |l|$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-inverse-zero*:

$\forall y::\text{real}. \exists N. \forall n \geq N. y < f(n)$   
 $\implies (\%n. \text{inverse}(f n)) \text{----} \text{NS} > 0$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-inverse-real-of-nat*:  $(\%n. \text{inverse}(\text{real}(\text{Suc } n))) \text{----} \text{NS} > 0$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-inverse-real-of-nat-add*:

$(\%n. r + \text{inverse}(\text{real}(\text{Suc } n))) \text{----} \text{NS} > r$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-inverse-real-of-nat-add-minus*:

$(\%n. r + -\text{inverse}(\text{real}(\text{Suc } n))) \text{----} \text{NS} > r$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-inverse-real-of-nat-add-minus-mult*:

$(\%n. r*(1 + \text{inverse}(\text{real}(\text{Suc } n)))) \text{----NS} > r$   
 ⟨proof⟩

## 12.2 Convergence

**lemma** *nslimI*:  $X \text{----NS} > L \implies \text{nslim } X = L$   
 ⟨proof⟩

**lemma** *lim-nslim-iff*:  $\text{lim } X = \text{nslim } X$   
 ⟨proof⟩

**lemma** *NSconvergentD*:  $\text{NSconvergent } X \implies \exists L. (X \text{----NS} > L)$   
 ⟨proof⟩

**lemma** *NSconvergentI*:  $(X \text{----NS} > L) \implies \text{NSconvergent } X$   
 ⟨proof⟩

**lemma** *convergent-NSconvergent-iff*:  $\text{convergent } X = \text{NSconvergent } X$   
 ⟨proof⟩

**lemma** *NSconvergent-NSLIMSEQ-iff*:  $\text{NSconvergent } X = (X \text{----NS} > \text{nslim } X)$   
 ⟨proof⟩

## 12.3 Bounded Monotonic Sequences

**lemma** *NSBseqD*:  $[\text{NSBseq } X; N : \text{HNatInfinite}] \implies (*f* X) N : \text{HFinite}$   
 ⟨proof⟩

**lemma** *Standard-subset-HFfinite*:  $\text{Standard} \subseteq \text{HFinite}$   
 ⟨proof⟩

**lemma** *NSBseqD2*:  $\text{NSBseq } X \implies (*f* X) N \in \text{HFinite}$   
 ⟨proof⟩

**lemma** *NSBseqI*:  $\forall N \in \text{HNatInfinite}. (*f* X) N : \text{HFinite} \implies \text{NSBseq } X$   
 ⟨proof⟩

The standard definition implies the nonstandard definition

**lemma** *Bseq-NSBseq*:  $\text{Bseq } X \implies \text{NSBseq } X$   
 ⟨proof⟩

The nonstandard definition implies the standard definition

**lemma** *SReal-less-omega*:  $r \in \mathbb{R} \implies r < \omega$   
 ⟨proof⟩

**lemma** *NSBseq-Bseq*:  $\text{NSBseq } X \implies \text{Bseq } X$   
 ⟨proof⟩

Equivalence of nonstandard and standard definitions for a bounded sequence

**lemma** *Bseq-NSBseq-iff*:  $(Bseq\ X) = (NSBseq\ X)$   
 ⟨proof⟩

A convergent sequence is bounded: Boundedness as a necessary condition for convergence. The nonstandard version has no existential, as usual

**lemma** *NSconvergent-NSBseq*:  $NSconvergent\ X ==> NSBseq\ X$   
 ⟨proof⟩

Standard Version: easily now proved using equivalence of NS and standard definitions

**lemma** *convergent-Bseq*:  $convergent\ X ==> Bseq\ X$   
 ⟨proof⟩

### 12.3.1 Upper Bounds and Lubs of Bounded Sequences

**lemma** *NSBseq-isUb*:  $NSBseq\ X ==> \exists U::real. isUb\ UNIV\ \{x. \exists n. X\ n = x\}$   
 $U$   
 ⟨proof⟩

**lemma** *NSBseq-isLub*:  $NSBseq\ X ==> \exists U::real. isLub\ UNIV\ \{x. \exists n. X\ n = x\}$   
 $U$   
 ⟨proof⟩

### 12.3.2 A Bounded and Monotonic Sequence Converges

The best of both worlds: Easier to prove this result as a standard theorem and then use equivalence to ”transfer” it into the equivalent nonstandard form if needed!

**lemma** *Bmonoseq-NSLIMSEQ*:  $\forall n \geq m. X\ n = X\ m ==> \exists L. (X\ \text{----NS}\ > L)$   
 ⟨proof⟩

**lemma** *NSBseq-mono-NSconvergent*:  
 $[[\ NSBseq\ X; \forall m. \forall n \geq m. X\ m \leq X\ n ]] ==> NSconvergent\ (X::nat=>real)$   
 ⟨proof⟩

## 12.4 Cauchy Sequences

**lemma** *NSCauchyI*:  
 $(\bigwedge M\ N. [[M \in HNatInfinite; N \in HNatInfinite]] \implies starfun\ X\ M \approx starfun\ X\ N)$   
 $\implies NSCauchy\ X$   
 ⟨proof⟩

**lemma** *NSCauchyD*:  
 $[[NSCauchy\ X; M \in HNatInfinite; N \in HNatInfinite]]$   
 $\implies starfun\ X\ M \approx starfun\ X\ N$   
 ⟨proof⟩

### 12.4.1 Equivalence Between NS and Standard

**lemma** *Cauchy-NSCauchy*:

**assumes**  $X$ : *Cauchy*  $X$  **shows** *NSCauchy*  $X$   
*<proof>*

**lemma** *NSCauchy-Cauchy*:

**assumes**  $X$ : *NSCauchy*  $X$  **shows** *Cauchy*  $X$   
*<proof>*

**theorem** *NSCauchy-Cauchy-iff*: *NSCauchy*  $X =$  *Cauchy*  $X$

*<proof>*

### 12.4.2 Cauchy Sequences are Bounded

A Cauchy sequence is bounded – nonstandard version

**lemma** *NSCauchy-NSBseq*: *NSCauchy*  $X ==>$  *NSBseq*  $X$

*<proof>*

### 12.4.3 Cauchy Sequences are Convergent

Equivalence of Cauchy criterion and convergence: We will prove this using our NS formulation which provides a much easier proof than using the standard definition. We do not need to use properties of subsequences such as boundedness, monotonicity etc... Compare with Harrison’s corresponding proof in HOL which is much longer and more complicated. Of course, we do not have problems which he encountered with guessing the right instantiations for his ‘epsilon-delta’ proof(s) in this case since the NS formulations do not involve existential quantifiers.

**lemma** *NSconvergent-NSCauchy*: *NSconvergent*  $X ==>$  *NSCauchy*  $X$

*<proof>*

**lemma** *real-NSCauchy-NSconvergent*:

**fixes**  $X$  :: *nat*  $\Rightarrow$  *real*  
**shows** *NSCauchy*  $X ==>$  *NSconvergent*  $X$   
*<proof>*

**lemma** *NSCauchy-NSconvergent*:

**fixes**  $X$  :: *nat*  $\Rightarrow$  '*a*::*banach*  
**shows** *NSCauchy*  $X ==>$  *NSconvergent*  $X$   
*<proof>*

**lemma** *NSCauchy-NSconvergent-iff*:

**fixes**  $X$  :: *nat*  $\Rightarrow$  '*a*::*banach*  
**shows** *NSCauchy*  $X =$  *NSconvergent*  $X$   
*<proof>*

## 12.5 Power Sequences

The sequence  $x^n$  tends to 0 if  $(0::'a) \leq x$  and  $x < (1::'a)$ . Proof will use (NS) Cauchy equivalence for convergence and also fact that bounded and monotonic sequence converges.

We now use NS criterion to bring proof of theorem through

**lemma** *NSLIMSEQ-realpow-zero*:

$[| 0 \leq (x::real); x < 1 |] \implies (\%n. x ^ n) \text{ ----NS} > 0$   
 $\langle proof \rangle$

**lemma** *NSLIMSEQ-rabs-realpow-zero*:  $|c| < (1::real) \implies (\%n. |c| ^ n) \text{ ----NS} > 0$

$\langle proof \rangle$

**lemma** *NSLIMSEQ-rabs-realpow-zero2*:  $|c| < (1::real) \implies (\%n. c ^ n) \text{ ----NS} > 0$

$\langle proof \rangle$

end

## 13 HSeries: Finite Summation and Infinite Series for Hyperreals

**theory** *HSeries*

**imports** *Series HSEQ*

**begin**

**definition**

$sumhr :: (hypnat * hypnat * (nat \Rightarrow real)) \Rightarrow hypreal$  **where**  
 $sumhr =$   
 $(\%(M,N,f). starfun2 (\%m n. setsum f \{m..<n\}) M N)$

**definition**

$NSsums :: [nat \Rightarrow real, real] \Rightarrow bool$  (**infixr** *NSsums* 80) **where**  
 $f NSsums s = (\%n. setsum f \{0..<n\}) \text{ ----NS} > s$

**definition**

$NSsummable :: (nat \Rightarrow real) \Rightarrow bool$  **where**  
 $NSsummable f = (\exists s. f NSsums s)$

**definition**

$NSsuminf :: (nat \Rightarrow real) \Rightarrow real$  **where**  
 $NSsuminf f = (THE s. f NSsums s)$

**lemma** *sumhr-app*:  $sumhr(M,N,f) = (*f2* (\lambda m n. setsum f \{m..<n\})) M N$   
 ⟨proof⟩

Base case in definition of *sumr*

**lemma** *sumhr-zero* [*simp*]:  $!!m. sumhr(m,0,f) = 0$   
 ⟨proof⟩

Recursive case in definition of *sumr*

**lemma** *sumhr-if*:  
 $!!m n. sumhr(m,n+1,f) =$   
 $(if n + 1 \leq m then 0 else sumhr(m,n,f) + (*f* f) n)$   
 ⟨proof⟩

**lemma** *sumhr-Suc-zero* [*simp*]:  $!!n. sumhr(n + 1, n, f) = 0$   
 ⟨proof⟩

**lemma** *sumhr-eq-bounds* [*simp*]:  $!!n. sumhr(n,n,f) = 0$   
 ⟨proof⟩

**lemma** *sumhr-Suc* [*simp*]:  $!!m. sumhr(m,m + 1,f) = (*f* f) m$   
 ⟨proof⟩

**lemma** *sumhr-add-lbound-zero* [*simp*]:  $!!k m. sumhr(m+k,k,f) = 0$   
 ⟨proof⟩

**lemma** *sumhr-add*:  
 $!!m n. sumhr(m,n,f) + sumhr(m,n,g) = sumhr(m,n,%i. f i + g i)$   
 ⟨proof⟩

**lemma** *sumhr-mult*:  
 $!!m n. hypreal-of-real r * sumhr(m,n,f) = sumhr(m,n,%n. r * f n)$   
 ⟨proof⟩

**lemma** *sumhr-split-add*:  
 $!!n p. n < p ==> sumhr(0,n,f) + sumhr(n,p,f) = sumhr(0,p,f)$   
 ⟨proof⟩

**lemma** *sumhr-split-diff*:  $n < p ==> sumhr(0,p,f) - sumhr(0,n,f) = sumhr(n,p,f)$   
 ⟨proof⟩

**lemma** *sumhr-hrabs*:  $!!m n. abs(sumhr(m,n,f)) \leq sumhr(m,n,%i. abs(f i))$   
 ⟨proof⟩

other general version also needed

**lemma** *sumhr-fun-hypnat-eq*:  
 $(\forall r. m \leq r \ \& \ r < n \ --> f r = g r) \ -->$   
 $sumhr(hypnat-of-nat m, hypnat-of-nat n, f) =$   
 $sumhr(hypnat-of-nat m, hypnat-of-nat n, g)$   
 ⟨proof⟩

**lemma** *sumhr-const*:

!! $n$ .  $sumhr(0, n, \%i. r) = hypreal-of-hypnat\ n * hypreal-of-real\ r$   
 <proof>

**lemma** *sumhr-less-bounds-zero* [simp]: !! $m\ n$ .  $n < m ==> sumhr(m, n, f) = 0$   
 <proof>

**lemma** *sumhr-minus*: !! $m\ n$ .  $sumhr(m, n, \%i. - f\ i) = - sumhr(m, n, f)$   
 <proof>

**lemma** *sumhr-shift-bounds*:

!! $m\ n$ .  $sumhr(m+hypnat-of-nat\ k, n+hypnat-of-nat\ k, f) =$   
 $sumhr(m, n, \%i. f(i + k))$   
 <proof>

### 13.1 Nonstandard Sums

Infinite sums are obtained by summing to some infinite hypernatural (such as *whn*)

**lemma** *sumhr-hypreal-of-hypnat-omega*:

$sumhr(0, whn, \%i. 1) = hypreal-of-hypnat\ whn$   
 <proof>

**lemma** *sumhr-hypreal-omega-minus-one*:  $sumhr(0, whn, \%i. 1) = omega - 1$   
 <proof>

**lemma** *sumhr-minus-one-realpow-zero* [simp]:

!! $N$ .  $sumhr(0, N + N, \%i. (-1) ^ (i+1)) = 0$   
 <proof>

**lemma** *sumhr-interval-const*:

( $\forall n. m \leq Suc\ n \rightarrow f\ n = r$ ) &  $m \leq na$   
 $==> sumhr(hypnat-of-nat\ m, hypnat-of-nat\ na, f) =$   
 $(hypreal-of-nat\ (na - m) * hypreal-of-real\ r)$   
 <proof>

**lemma** *starfunNat-sumr*: !! $N$ . ( $*f*$  ( $\%n. setsum\ f\ \{0..<n\}$ ))  $N = sumhr(0, N, f)$   
 <proof>

**lemma** *sumhr-hrabs-approx* [simp]:  $sumhr(0, M, f) @= sumhr(0, N, f)$

$==> abs\ (sumhr(M, N, f)) @= 0$   
 <proof>

**lemma** *sums-NSsums-iff*:  $(f\ sums\ l) = (f\ NSsums\ l)$

<proof>

**lemma** *summable-NSsummable-iff*:  $(summable\ f) = (NSsummable\ f)$

*<proof>*

**lemma** *suminf-NSsuminf-iff*:  $(\text{suminf } f) = (\text{NSsuminf } f)$   
*<proof>*

**lemma** *NSsums-NSsummable*:  $f \text{ NSsums } l \implies \text{NSsummable } f$   
*<proof>*

**lemma** *NSsummable-NSsums*:  $\text{NSsummable } f \implies f \text{ NSsums } (\text{NSsuminf } f)$   
*<proof>*

**lemma** *NSsums-unique*:  $f \text{ NSsums } s \implies (s = \text{NSsuminf } f)$   
*<proof>*

**lemma** *NSseries-zero*:  
 $\forall m. n \leq \text{Suc } m \implies f(m) = 0 \implies f \text{ NSsums } (\text{setsum } f \{0..<n\})$   
*<proof>*

**lemma** *NSsummable-NSCauchy*:  
 $\text{NSsummable } f =$   
 $(\forall M \in \text{HNatInfinite}. \forall N \in \text{HNatInfinite}. \text{abs } (\text{sumhr}(M,N,f)) \text{ @} = 0)$   
*<proof>*

Terms of a convergent series tend to zero

**lemma** *NSsummable-NSLIMSEQ-zero*:  $\text{NSsummable } f \implies f \text{ ----NS} > 0$   
*<proof>*

Nonstandard comparison test

**lemma** *NSsummable-comparison-test*:  
 $[\exists N. \forall n. N \leq n \implies \text{abs}(f n) \leq g n; \text{NSsummable } g] \implies \text{NSsummable } f$   
*<proof>*

**lemma** *NSsummable-rabs-comparison-test*:  
 $[\exists N. \forall n. N \leq n \implies \text{abs}(f n) \leq g n; \text{NSsummable } g] \implies \text{NSsummable } (\%k. \text{abs } (f k))$   
*<proof>*

**end**

## 14 HLim: Limits and Continuity (Nonstandard)

**theory** *HLim*  
**imports** *Star Lim*  
**begin**

Nonstandard Definitions

**definition**

$NSLIM :: [ 'a::real-normed-vector \Rightarrow 'b::real-normed-vector, 'a, 'b ] \Rightarrow bool$   
 $(((-)/ \text{---} (-)/ \text{---} NS > (-)) [60, 0, 60] 60)$  **where**  
 $f \text{---} a \text{---} NS > L =$   
 $(\forall x. (x \neq \text{star-of } a \ \& \ x @ = \text{star-of } a \text{---} \rightarrow ( *f* f ) x @ = \text{star-of } L))$

**definition**

$isNSCont :: [ 'a::real-normed-vector \Rightarrow 'b::real-normed-vector, 'a ] \Rightarrow bool$  **where**  
 $\text{---} NS$  definition dispenses with limit notions  
 $isNSCont f a = (\forall y. y @ = \text{star-of } a \text{---} \rightarrow$   
 $( *f* f ) y @ = \text{star-of } (f a))$

**definition**

$isNSUCont :: [ 'a::real-normed-vector \Rightarrow 'b::real-normed-vector ] \Rightarrow bool$  **where**  
 $isNSUCont f = (\forall x y. x @ = y \text{---} \rightarrow ( *f* f ) x @ = ( *f* f ) y)$

**14.1 Limits of Functions****lemma NSLIM-I:**

$(\bigwedge x. [x \neq \text{star-of } a; x \approx \text{star-of } a] \implies \text{starfun } f x \approx \text{star-of } L)$   
 $\implies f \text{---} a \text{---} NS > L$   
 $\langle \text{proof} \rangle$

**lemma NSLIM-D:**

$[f \text{---} a \text{---} NS > L; x \neq \text{star-of } a; x \approx \text{star-of } a]$   
 $\implies \text{starfun } f x \approx \text{star-of } L$   
 $\langle \text{proof} \rangle$

Proving properties of limits using nonstandard definition. The properties hold for standard limits as well!

**lemma NSLIM-mult:**

**fixes**  $l m :: 'a::real-normed-algebra$   
**shows**  $[ [ f \text{---} x \text{---} NS > l; g \text{---} x \text{---} NS > m ] ]$   
 $\implies (\%x. f(x) * g(x)) \text{---} x \text{---} NS > (l * m)$   
 $\langle \text{proof} \rangle$

**lemma starfun-scaleR [simp]:**

$\text{starfun } (\lambda x. f x *_R g x) = (\lambda x. \text{scaleHR } (\text{starfun } f x) (\text{starfun } g x))$   
 $\langle \text{proof} \rangle$

**lemma NSLIM-scaleR:**

$[ [ f \text{---} x \text{---} NS > l; g \text{---} x \text{---} NS > m ] ]$   
 $\implies (\%x. f(x) *_R g(x)) \text{---} x \text{---} NS > (l *_R m)$   
 $\langle \text{proof} \rangle$

**lemma NSLIM-add:**

$[ [ f \text{---} x \text{---} NS > l; g \text{---} x \text{---} NS > m ] ]$   
 $\implies (\%x. f(x) + g(x)) \text{---} x \text{---} NS > (l + m)$   
 $\langle \text{proof} \rangle$

**lemma** *NSLIM-const* [*simp*]:  $(\%x. k) \text{ -- } x \text{ --NS} > k$   
 ⟨*proof*⟩

**lemma** *NSLIM-minus*:  $f \text{ -- } a \text{ --NS} > L \implies (\%x. -f(x)) \text{ -- } a \text{ --NS} > -L$   
 ⟨*proof*⟩

**lemma** *NSLIM-diff*:  
 $\llbracket f \text{ -- } x \text{ --NS} > l; g \text{ -- } x \text{ --NS} > m \rrbracket \implies (\lambda x. f x - g x) \text{ -- } x \text{ --NS} > (l - m)$   
 ⟨*proof*⟩

**lemma** *NSLIM-add-minus*:  $\llbracket f \text{ -- } x \text{ --NS} > l; g \text{ -- } x \text{ --NS} > m \rrbracket \implies$   
 $(\%x. f(x) + -g(x)) \text{ -- } x \text{ --NS} > (l + -m)$   
 ⟨*proof*⟩

**lemma** *NSLIM-inverse*:  
**fixes**  $L :: 'a::\text{real-normed-div-algebra}$   
**shows**  $\llbracket f \text{ -- } a \text{ --NS} > L; L \neq 0 \rrbracket$   
 $\implies (\%x. \text{inverse}(f(x))) \text{ -- } a \text{ --NS} > (\text{inverse } L)$   
 ⟨*proof*⟩

**lemma** *NSLIM-zero*:  
**assumes**  $f: f \text{ -- } a \text{ --NS} > l$  **shows**  $(\%x. f(x) - l) \text{ -- } a \text{ --NS} > 0$   
 ⟨*proof*⟩

**lemma** *NSLIM-zero-cancel*:  $(\%x. f(x) - l) \text{ -- } x \text{ --NS} > 0 \implies f \text{ -- } x \text{ --NS} > l$   
 ⟨*proof*⟩

**lemma** *NSLIM-const-not-eq*:  
**fixes**  $a :: 'a::\text{real-normed-algebra-1}$   
**shows**  $k \neq L \implies \neg (\lambda x. k) \text{ -- } a \text{ --NS} > L$   
 ⟨*proof*⟩

**lemma** *NSLIM-not-zero*:  
**fixes**  $a :: 'a::\text{real-normed-algebra-1}$   
**shows**  $k \neq 0 \implies \neg (\lambda x. k) \text{ -- } a \text{ --NS} > 0$   
 ⟨*proof*⟩

**lemma** *NSLIM-const-eq*:  
**fixes**  $a :: 'a::\text{real-normed-algebra-1}$   
**shows**  $(\lambda x. k) \text{ -- } a \text{ --NS} > L \implies k = L$   
 ⟨*proof*⟩

**lemma** *NSLIM-unique*:  
**fixes**  $a :: 'a::\text{real-normed-algebra-1}$   
**shows**  $\llbracket f \text{ -- } a \text{ --NS} > L; f \text{ -- } a \text{ --NS} > M \rrbracket \implies L = M$   
 ⟨*proof*⟩

**lemma** *NSLIM-mult-zero*:

**fixes**  $f\ g :: 'a::\text{real-normed-vector} \Rightarrow 'b::\text{real-normed-algebra}$   
**shows**  $\llbracket f \dashrightarrow x \dashrightarrow NS > 0; g \dashrightarrow x \dashrightarrow NS > 0 \rrbracket \implies (\%x. f(x)*g(x)) \dashrightarrow x \dashrightarrow NS > 0$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIM-self*:  $(\%x. x) \dashrightarrow a \dashrightarrow NS > a$   
 $\langle\text{proof}\rangle$

### 14.1.1 Equivalence of LIM and NSLIM

**lemma** *LIM-NSLIM*:

**assumes**  $f: f \dashrightarrow a \dashrightarrow L$  **shows**  $f \dashrightarrow a \dashrightarrow NS > L$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIM-LIM*:

**assumes**  $f: f \dashrightarrow a \dashrightarrow NS > L$  **shows**  $f \dashrightarrow a \dashrightarrow L$   
 $\langle\text{proof}\rangle$

**theorem** *LIM-NSLIM-iff*:  $(f \dashrightarrow x \dashrightarrow L) = (f \dashrightarrow x \dashrightarrow NS > L)$   
 $\langle\text{proof}\rangle$

## 14.2 Continuity

**lemma** *isNSContD*:

$\llbracket \text{isNSCont } f\ a; y \approx \text{star-of } a \rrbracket \implies (*f* f)\ y \approx \text{star-of } (f\ a)$   
 $\langle\text{proof}\rangle$

**lemma** *isNSCont-NSLIM*:  $\text{isNSCont } f\ a \implies f \dashrightarrow a \dashrightarrow NS > (f\ a)$   
 $\langle\text{proof}\rangle$

**lemma** *NSLIM-isNSCont*:  $f \dashrightarrow a \dashrightarrow NS > (f\ a) \implies \text{isNSCont } f\ a$   
 $\langle\text{proof}\rangle$

NS continuity can be defined using NS Limit in similar fashion to standard def of continuity

**lemma** *isNSCont-NSLIM-iff*:  $(\text{isNSCont } f\ a) = (f \dashrightarrow a \dashrightarrow NS > (f\ a))$   
 $\langle\text{proof}\rangle$

Hence, NS continuity can be given in terms of standard limit

**lemma** *isNSCont-LIM-iff*:  $(\text{isNSCont } f\ a) = (f \dashrightarrow a \dashrightarrow (f\ a))$   
 $\langle\text{proof}\rangle$

Moreover, it's trivial now that NS continuity is equivalent to standard continuity

**lemma** *isNSCont-isCont-iff*:  $(\text{isNSCont } f\ a) = (\text{isCont } f\ a)$   
 $\langle\text{proof}\rangle$

Standard continuity  $\equiv_i$  NS continuity

**lemma** *isCont-isNSCont*:  $isCont\ f\ a \implies isNSCont\ f\ a$   
 $\langle proof \rangle$

NS continuity  $\equiv_i$  Standard continuity

**lemma** *isNSCont-isCont*:  $isNSCont\ f\ a \implies isCont\ f\ a$   
 $\langle proof \rangle$

Alternative definition of continuity

**lemma** *NSLIM-h-iff*:  $(f \dashv\vdash a \dashv\vdash NS > L) = ((\%h. f(a + h)) \dashv\vdash 0 \dashv\vdash NS > L)$   
 $\langle proof \rangle$

**lemma** *NSLIM-isCont-iff*:  $(f \dashv\vdash a \dashv\vdash NS > f\ a) = ((\%h. f(a + h)) \dashv\vdash 0 \dashv\vdash NS > f\ a)$   
 $\langle proof \rangle$

**lemma** *isNSCont-minus*:  $isNSCont\ f\ a \implies isNSCont\ (\%x. - f\ x)\ a$   
 $\langle proof \rangle$

**lemma** *isNSCont-inverse*:

**fixes**  $f :: 'a::real-normed-vector \Rightarrow 'b::real-normed-div-algebra$   
**shows**  $[\mid isNSCont\ f\ x; f\ x \neq 0 \mid] \implies isNSCont\ (\%x. inverse\ (f\ x))\ x$   
 $\langle proof \rangle$

**lemma** *isNSCont-const [simp]*:  $isNSCont\ (\%x. k)\ a$   
 $\langle proof \rangle$

**lemma** *isNSCont-abs [simp]*:  $isNSCont\ abs\ (a::real)$   
 $\langle proof \rangle$

### 14.3 Uniform Continuity

**lemma** *isNSUContD*:  $[\mid isNSUCont\ f; x \approx y \mid] \implies (*f* f)\ x \approx (*f* f)\ y$   
 $\langle proof \rangle$

**lemma** *isUCont-isNSUCont*:

**fixes**  $f :: 'a::real-normed-vector \Rightarrow 'b::real-normed-vector$   
**assumes**  $f: isUCont\ f$  **shows**  $isNSUCont\ f$   
 $\langle proof \rangle$

**lemma** *isNSUCont-isUCont*:

**fixes**  $f :: 'a::real-normed-vector \Rightarrow 'b::real-normed-vector$   
**assumes**  $f: isNSUCont\ f$  **shows**  $isUCont\ f$   
 $\langle proof \rangle$

**end**

## 15 HDeriv: Differentiation (Nonstandard)

```
theory HDeriv
imports Deriv HLim
begin
```

Nonstandard Definitions

**definition**

```
nsderiv :: ['a::real-normed-field  $\Rightarrow$  'a, 'a, 'a]  $\Rightarrow$  bool
  ((NSDERIV (-) / (-) / :> (-)) [1000, 1000, 60] 60) where
NSDERIV f x :> D = ( $\forall$  h  $\in$  Infinitesimal - {0}.
  (( *f* f)(star-of x + h)
  - star-of (f x))/h @= star-of D)
```

**definition**

```
NSdifferentiable :: ['a::real-normed-field  $\Rightarrow$  'a, 'a]  $\Rightarrow$  bool
  (infixl NSdifferentiable 60) where
f NSdifferentiable x = ( $\exists$  D. NSDERIV f x :> D)
```

**definition**

```
increment :: [real= $\Rightarrow$ real,real,hypreal]  $\Rightarrow$  hypreal where
increment f x h = (@inc. f NSdifferentiable x &
  inc = ( *f* f)(hypreal-of-real x + h) - hypreal-of-real (f x))
```

### 15.1 Derivatives

**lemma** DERIV-NS-iff:

```
(DERIV f x :> D) = ((%h. (f(x + h) - f(x))/h) -- 0 --NS> D)
<proof>
```

**lemma** NS-DERIV-D: DERIV f x :> D  $\implies$  (%h. (f(x + h) - f(x))/h) -- 0 --NS> D

<proof>

**lemma** hnorm-of-hypreal:

```
 $\bigwedge$ r. hnorm (( *f* of-real) r::'a::real-normed-div-algebra star) = |r|
<proof>
```

**lemma** Infinitesimal-of-hypreal:

```
x  $\in$  Infinitesimal  $\implies$ 
  (( *f* of-real) x::'a::real-normed-div-algebra star)  $\in$  Infinitesimal
<proof>
```

**lemma** of-hypreal-eq-0-iff:

```
 $\bigwedge$ x. (( *f* of-real) x = (0::'a::real-algebra-1 star)) = (x = 0)
<proof>
```

**lemma** NSDeriv-unique:

```
[| NSDERIV f x :> D; NSDERIV f x :> E |]  $\implies$  D = E
<proof>
```

First NSDERIV in terms of NSLIM

first equivalence

**lemma** *NSDERIV-NSLIM-iff:*

$$(NSDERIV f x :> D) = ((\%h. (f(x + h) - f(x))/h) \text{--- } 0 \text{---} NS > D)$$

*<proof>*

second equivalence

**lemma** *NSDERIV-NSLIM-iff2:*

$$(NSDERIV f x :> D) = ((\%z. (f(z) - f(x)) / (z-x)) \text{--- } x \text{---} NS > D)$$

*<proof>*

**lemma** *NSDERIV-iff2:*

$$(NSDERIV f x :> D) =$$

$$(\forall w.$$

$$w \neq \text{star-of } x \ \& \ w \approx \text{star-of } x \text{---} >$$

$$(\text{*f* } (\%z. (f z - f x) / (z-x))) \ w \approx \text{star-of } D)$$

*<proof>*

**lemma** *hypreal-not-eq-minus-iff:*

$$(x \neq a) = (x - a \neq (0::'a::\text{ab-group-add}))$$

*<proof>*

**lemma** *NSDERIVD5:*

$$(NSDERIV f x :> D) ==>$$

$$(\forall u. u \approx \text{hypreal-of-real } x \text{---} >$$

$$(\text{*f* } (\%z. f z - f x)) \ u \approx \text{hypreal-of-real } D * (u - \text{hypreal-of-real } x))$$

*<proof>*

**lemma** *NSDERIVD4:*

$$(NSDERIV f x :> D) ==>$$

$$(\forall h \in \text{Infinitesimal.}$$

$$((\text{*f* } f)(\text{hypreal-of-real } x + h) -$$

$$\text{hypreal-of-real } (f x)) \approx (\text{hypreal-of-real } D) * h)$$

*<proof>*

**lemma** *NSDERIVD3:*

$$(NSDERIV f x :> D) ==>$$

$$(\forall h \in \text{Infinitesimal} - \{0\}.$$

$$((\text{*f* } f)(\text{hypreal-of-real } x + h) -$$

$$\text{hypreal-of-real } (f x)) \approx (\text{hypreal-of-real } D) * h)$$

*<proof>*

Differentiability implies continuity nice and simple "algebraic" proof

**lemma** *NSDERIV-isNSCont:*  $NSDERIV f x :> D ==> \text{isNSCont } f x$

*<proof>*

Differentiation rules for combinations of functions follow from clear, straightforward, algebraic manipulations

Constant function

**lemma** *NSDERIV-const* [*simp*]: (*NSDERIV* (%*x*. *k*) *x* := 0)  
 ⟨*proof*⟩

Sum of functions- proved easily

**lemma** *NSDERIV-add*: [| *NSDERIV f x* := *Da*; *NSDERIV g x* := *Db* |]  
 ==> *NSDERIV* (%*x*. *f x* + *g x*) *x* := *Da* + *Db*  
 ⟨*proof*⟩

Product of functions - Proof is trivial but tedious and long due to rearrangement of terms

**lemma** *lemma-nsderiv1*:  
**fixes** *a b c d* :: '*a*::*comm-ring star*  
**shows** (*a\*b*) - (*c\*d*) = (*b\*(a - c)*) + (*c\*(b - d)*)  
 ⟨*proof*⟩

**lemma** *lemma-nsderiv2*:  
**fixes** *x y z* :: '*a*::*real-normed-field star*  
**shows** [| (*x - y*) / *z* = *star-of D* + *yb*; *z* ≠ 0;  
*z* ∈ *Infinitesimal*; *yb* ∈ *Infinitesimal* |]  
 ==> *x - y* ≈ 0  
 ⟨*proof*⟩

**lemma** *NSDERIV-mult*: [| *NSDERIV f x* := *Da*; *NSDERIV g x* := *Db* |]  
 ==> *NSDERIV* (%*x*. *f x* \* *g x*) *x* := (*Da* \* *g(x)*) + (*Db* \* *f(x)*)  
 ⟨*proof*⟩

Multiplying by a constant

**lemma** *NSDERIV-cmult*: *NSDERIV f x* := *D*  
 ==> *NSDERIV* (%*x*. *c* \* *f x*) *x* := *c\*D*  
 ⟨*proof*⟩

Negation of function

**lemma** *NSDERIV-minus*: *NSDERIV f x* := *D* ==> *NSDERIV* (%*x*. -(*f x*)) *x*  
 := -*D*  
 ⟨*proof*⟩

Subtraction

**lemma** *NSDERIV-add-minus*: [| *NSDERIV f x* := *Da*; *NSDERIV g x* := *Db* |]  
 ==> *NSDERIV* (%*x*. *f x* + -*g x*) *x* := *Da* + -*Db*  
 ⟨*proof*⟩

**lemma** *NSDERIV-diff*:  
 [| *NSDERIV f x* := *Da*; *NSDERIV g x* := *Db* |]

$\implies \text{NSDERIV } (\%x. f\ x - g\ x)\ x \text{ :> } Da - Db$   
 <proof>

Similarly to the above, the chain rule admits an entirely straightforward derivation. Compare this with Harrison’s HOL proof of the chain rule, which proved to be trickier and required an alternative characterisation of differentiability- the so-called Carathedory derivative. Our main problem is manipulation of terms.

**lemma NSDERIV-zero:**

$\llbracket \text{NSDERIV } g\ x \text{ :> } D;$   
 $(\ *f*\ g)\ (\text{star-of } x + xa) = \text{star-of } (g\ x);$   
 $xa \in \text{Infinitesimal};$   
 $xa \neq 0$   
 $\rrbracket \implies D = 0$   
 <proof>

**lemma NSDERIV-approx:**

$\llbracket \text{NSDERIV } f\ x \text{ :> } D; h \in \text{Infinitesimal}; h \neq 0 \rrbracket$   
 $\implies (\ *f*\ f)\ (\text{star-of } x + h) - \text{star-of } (f\ x) \approx 0$   
 <proof>

**lemma NSDERIVD1:**  $\llbracket \text{NSDERIV } f\ (g\ x) \text{ :> } Da;$

$(\ *f*\ g)\ (\text{star-of } (x) + xa) \neq \text{star-of } (g\ x);$   
 $(\ *f*\ g)\ (\text{star-of } (x) + xa) \approx \text{star-of } (g\ x)$   
 $\rrbracket \implies ((\ *f*\ f)\ ((\ *f*\ g)\ (\text{star-of } (x) + xa))$   
 $\quad - \text{star-of } (f\ (g\ x)))$   
 $\quad / ((\ *f*\ g)\ (\text{star-of } (x) + xa) - \text{star-of } (g\ x))$   
 $\approx \text{star-of } (Da)$   
 <proof>

**lemma NSDERIVD2:**  $\llbracket \text{NSDERIV } g\ x \text{ :> } Db; xa \in \text{Infinitesimal}; xa \neq 0 \rrbracket$

$\implies ((\ *f*\ g)\ (\text{star-of } (x) + xa) - \text{star-of } (g\ x)) / xa$   
 $\approx \text{star-of } (Db)$   
 <proof>

**lemma lemma-chain:**  $(z::'a::\text{real-normed-field star}) \neq 0 \implies x*y = (x*\text{inverse}(z))*(z*y)$   
 <proof>

This proof uses both definitions of differentiability.

**lemma NSDERIV-chain:**  $\llbracket \text{NSDERIV } f\ (g\ x) \text{ :> } Da; \text{NSDERIV } g\ x \text{ :> } Db \rrbracket$

$\implies \text{NSDERIV } (f\ o\ g)\ x \text{ :> } Da * Db$   
 <proof>

Differentiation of natural number powers

**lemma** *NSDERIV-Id* [simp]:  $NSDERIV (\%x. x) x :=> 1$   
 ⟨proof⟩

**lemma** *NSDERIV-cmult-Id* [simp]:  $NSDERIV (op * c) x :=> c$   
 ⟨proof⟩

**lemma** *NSDERIV-inverse*:  
 fixes  $x :: 'a::\{real-normed-field\}$   
 shows  $x \neq 0 \implies NSDERIV (\%x. inverse(x)) x :=> (- (inverse x ^ Suc (Suc 0)))$   
 ⟨proof⟩

### 15.1.1 Equivalence of NS and Standard definitions

**lemma** *divideR-eq-divide*:  $x /_R y = x / y$   
 ⟨proof⟩

Now equivalence between NSDERIV and DERIV

**lemma** *NSDERIV-DERIV-iff*:  $(NSDERIV f x :=> D) = (DERIV f x :=> D)$   
 ⟨proof⟩

**lemma** *NSDERIV-pow*:  $NSDERIV (\%x. x ^ n) x :=> real n * (x ^ (n - Suc 0))$   
 ⟨proof⟩

Derivative of inverse

**lemma** *NSDERIV-inverse-fun*:  
 fixes  $x :: 'a::\{real-normed-field\}$   
 shows  $[\![ NSDERIV f x :=> d; f(x) \neq 0 ]\!] \implies NSDERIV (\%x. inverse(f x)) x :=> (- (d * inverse(f x) ^ Suc (Suc 0)))$   
 ⟨proof⟩

Derivative of quotient

**lemma** *NSDERIV-quotient*:  
 fixes  $x :: 'a::\{real-normed-field\}$   
 shows  $[\![ NSDERIV f x :=> d; NSDERIV g x :=> e; g(x) \neq 0 ]\!] \implies NSDERIV (\%y. f(y) / (g y)) x :=> (d*g(x) - (e*f(x))) / (g(x) ^ Suc (Suc 0))$   
 ⟨proof⟩

**lemma** *CARAT-NSDERIV*:  $NSDERIV f x :=> l \implies \exists g. (\forall z. f z - f x = g z * (z-x)) \ \& \ isNSCont g x \ \& \ g x = l$   
 ⟨proof⟩

**lemma** *hypreal-eq-minus-iff3*:  $(x = y + z) = (x + -z = (y::hypreal))$   
 ⟨proof⟩

**lemma** *CARAT-DERIVD*:  
**assumes** *all*:  $\forall z. f z - f x = g z * (z-x)$   
**and** *nsc*: *isNSCont* *g x*  
**shows** *NSDERIV* *f x*  $:>$  *g x*  
 $\langle$ *proof* $\rangle$

### 15.1.2 Differentiability predicate

**lemma** *NSdifferentiableD*: *f NSdifferentiable x*  $\implies \exists D. NSDERIV f x :> D$   
 $\langle$ *proof* $\rangle$

**lemma** *NSdifferentiableI*: *NSDERIV f x :> D*  $\implies f NSdifferentiable x$   
 $\langle$ *proof* $\rangle$

## 15.2 (NS) Increment

**lemma** *incrementI*:  
*f NSdifferentiable x*  $\implies$   
 $increment f x h = (*f* f) (hypreal-of-real(x) + h) -$   
 $hypreal-of-real (f x)$   
 $\langle$ *proof* $\rangle$

**lemma** *incrementI2*: *NSDERIV f x :> D*  $\implies$   
 $increment f x h = (*f* f) (hypreal-of-real(x) + h) -$   
 $hypreal-of-real (f x)$   
 $\langle$ *proof* $\rangle$

**lemma** *increment-thm*:  $[| NSDERIV f x :> D; h \in Infinitesimal; h \neq 0 |]$   
 $\implies \exists e \in Infinitesimal. increment f x h = hypreal-of-real(D)*h + e*h$   
 $\langle$ *proof* $\rangle$

**lemma** *increment-thm2*:  
 $[| NSDERIV f x :> D; h \approx 0; h \neq 0 |]$   
 $\implies \exists e \in Infinitesimal. increment f x h =$   
 $hypreal-of-real(D)*h + e*h$   
 $\langle$ *proof* $\rangle$

**lemma** *increment-approx-zero*:  $[| NSDERIV f x :> D; h \approx 0; h \neq 0 |]$   
 $\implies increment f x h \approx 0$   
 $\langle$ *proof* $\rangle$

**end**

## 16 HTranscendental: Nonstandard Extensions of Transcendental Functions

```
theory HTranscendental
imports Transcendental HSeries HDeriv
begin
```

### definition

```
exphr :: real => hypreal where
  — define exponential function using standard part
  exphr x = st(sumhr (0, whn, %n. inverse(real (fact n)) * (x ^ n)))
```

### definition

```
sinhr :: real => hypreal where
  sinhr x = st(sumhr (0, whn, %n. sin-coeff n * x ^ n))
```

### definition

```
coshhr :: real => hypreal where
  coshhr x = st(sumhr (0, whn, %n. cos-coeff n * x ^ n))
```

### 16.1 Nonstandard Extension of Square Root Function

```
lemma STAR-sqrt-zero [simp]: (*f* sqrt) 0 = 0
⟨proof⟩
```

```
lemma STAR-sqrt-one [simp]: (*f* sqrt) 1 = 1
⟨proof⟩
```

```
lemma hypreal-sqrt-pow2-iff: (( *f* sqrt)(x) ^ 2 = x) = (0 ≤ x)
⟨proof⟩
```

```
lemma hypreal-sqrt-gt-zero-pow2: !!x. 0 < x ==> ( *f* sqrt) (x) ^ 2 = x
⟨proof⟩
```

```
lemma hypreal-sqrt-pow2-gt-zero: 0 < x ==> 0 < ( *f* sqrt) (x) ^ 2
⟨proof⟩
```

```
lemma hypreal-sqrt-not-zero: 0 < x ==> ( *f* sqrt) (x) ≠ 0
⟨proof⟩
```

```
lemma hypreal-inverse-sqrt-pow2:
  0 < x ==> inverse (( *f* sqrt)(x)) ^ 2 = inverse x
⟨proof⟩
```

```
lemma hypreal-sqrt-mult-distrib:
  !!x y. [| 0 < x; 0 < y |] ==>
  ( *f* sqrt)(x*y) = ( *f* sqrt)(x) * ( *f* sqrt)(y)
⟨proof⟩
```

**lemma** *hypreal-sqrt-mult-distrib2*:

$$\begin{aligned} & [|0 \leq x; 0 \leq y|] ==> \\ & (*f* sqrt)(x*y) = (*f* sqrt)(x) * (*f* sqrt)(y) \\ & \langle proof \rangle \end{aligned}$$

**lemma** *hypreal-sqrt-approx-zero* [simp]:

$$0 < x ==> (( *f* sqrt)(x) @= 0) = (x @= 0)$$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-approx-zero2* [simp]:

$$0 \leq x ==> (( *f* sqrt)(x) @= 0) = (x @= 0)$$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-sum-squares* [simp]:

$$(( *f* sqrt)(x*x + y*y + z*z) @= 0) = (x*x + y*y + z*z @= 0)$$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-sum-squares2* [simp]:

$$(( *f* sqrt)(x*x + y*y) @= 0) = (x*x + y*y @= 0)$$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-gt-zero*:  $!!x. 0 < x ==> 0 < (*f* sqrt)(x)$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-ge-zero*:  $0 \leq x ==> 0 \leq (*f* sqrt)(x)$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-hrabs* [simp]:  $!!x. (*f* sqrt)(x ^ 2) = abs(x)$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-hrabs2* [simp]:  $!!x. (*f* sqrt)(x*x) = abs(x)$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-hyperpow-hrabs* [simp]:

$$!!x. (*f* sqrt)(x pow (hypnat-of-nat 2)) = abs(x)$$

$\langle proof \rangle$

**lemma** *star-sqrt-HFinite*:  $[|x \in HFinite; 0 \leq x|] ==> (*f* sqrt) x \in HFinite$

$\langle proof \rangle$

**lemma** *st-hypreal-sqrt*:

$$[|x \in HFinite; 0 \leq x|] ==> st((*f* sqrt) x) = (*f* sqrt)(st x)$$

$\langle proof \rangle$

**lemma** *hypreal-sqrt-sum-squares-ge1* [simp]:  $!!x y. x \leq (*f* sqrt)(x ^ 2 + y ^ 2)$

$\langle proof \rangle$

**lemma** *HFinite-hypreal-sqrt*:

$$[|0 \leq x; x \in HFinite|] ==> (*f* sqrt) x \in HFinite$$

⟨proof⟩

**lemma** *HFinite-hypreal-sqrt-imp-HFinite*:

$$\llbracket 0 \leq x; (*f* \text{ sqrt}) x \in \text{HFinite} \rrbracket \implies x \in \text{HFinite}$$

⟨proof⟩

**lemma** *HFinite-hypreal-sqrt-iff [simp]*:

$$0 \leq x \implies (( *f* \text{ sqrt}) x \in \text{HFinite}) = (x \in \text{HFinite})$$

⟨proof⟩

**lemma** *HFinite-sqrt-sum-squares [simp]*:

$$(( *f* \text{ sqrt})(x*x + y*y) \in \text{HFinite}) = (x*x + y*y \in \text{HFinite})$$

⟨proof⟩

**lemma** *Infinitesimal-hypreal-sqrt*:

$$\llbracket 0 \leq x; x \in \text{Infinitesimal} \rrbracket \implies (*f* \text{ sqrt}) x \in \text{Infinitesimal}$$

⟨proof⟩

**lemma** *Infinitesimal-hypreal-sqrt-imp-Infinitesimal*:

$$\llbracket 0 \leq x; (*f* \text{ sqrt}) x \in \text{Infinitesimal} \rrbracket \implies x \in \text{Infinitesimal}$$

⟨proof⟩

**lemma** *Infinitesimal-hypreal-sqrt-iff [simp]*:

$$0 \leq x \implies (( *f* \text{ sqrt}) x \in \text{Infinitesimal}) = (x \in \text{Infinitesimal})$$

⟨proof⟩

**lemma** *Infinitesimal-sqrt-sum-squares [simp]*:

$$(( *f* \text{ sqrt})(x*x + y*y) \in \text{Infinitesimal}) = (x*x + y*y \in \text{Infinitesimal})$$

⟨proof⟩

**lemma** *HInfinite-hypreal-sqrt*:

$$\llbracket 0 \leq x; x \in \text{HInfinite} \rrbracket \implies (*f* \text{ sqrt}) x \in \text{HInfinite}$$

⟨proof⟩

**lemma** *HInfinite-hypreal-sqrt-imp-HInfinite*:

$$\llbracket 0 \leq x; (*f* \text{ sqrt}) x \in \text{HInfinite} \rrbracket \implies x \in \text{HInfinite}$$

⟨proof⟩

**lemma** *HInfinite-hypreal-sqrt-iff [simp]*:

$$0 \leq x \implies (( *f* \text{ sqrt}) x \in \text{HInfinite}) = (x \in \text{HInfinite})$$

⟨proof⟩

**lemma** *HInfinite-sqrt-sum-squares [simp]*:

$$(( *f* \text{ sqrt})(x*x + y*y) \in \text{HInfinite}) = (x*x + y*y \in \text{HInfinite})$$

⟨proof⟩

**lemma** *HFinite-exp [simp]*:

$$\text{sumhr } (0, \text{whn}, \%n. \text{inverse (real (fact n))} * x \wedge n) \in \text{HFinite}$$

⟨proof⟩

**lemma** *exphr-zero* [*simp*]:  $\text{exphr } 0 = 1$   
 ⟨*proof*⟩

**lemma** *coshr-zero* [*simp*]:  $\text{coshr } 0 = 1$   
 ⟨*proof*⟩

**lemma** *STAR-exp-zero-approx-one* [*simp*]:  $(\text{*f* exp}) (0::\text{hypreal}) \text{@} = 1$   
 ⟨*proof*⟩

**lemma** *STAR-exp-Infinitesimal*:  $x \in \text{Infinitesimal} \implies (\text{*f* exp}) (x::\text{hypreal}) \text{@} = 1$   
 ⟨*proof*⟩

**lemma** *STAR-exp-epsilon* [*simp*]:  $(\text{*f* exp}) \text{epsilon} \text{@} = 1$   
 ⟨*proof*⟩

**lemma** *STAR-exp-add*:  $\forall x y. (\text{*f* exp})(x + y) = (\text{*f* exp}) x * (\text{*f* exp}) y$   
 ⟨*proof*⟩

**lemma** *exphr-hypreal-of-real-exp-eq*:  $\text{exphr } x = \text{hypreal-of-real } (\text{exp } x)$   
 ⟨*proof*⟩

**lemma** *starfun-exp-ge-add-one-self* [*simp*]:  $\forall x::\text{hypreal}. 0 \leq x \implies (1 + x) \leq (\text{*f* exp}) x$   
 ⟨*proof*⟩

**lemma** *starfun-exp-HInfinite*:  
 $\llbracket x \in \text{HInfinite}; 0 \leq x \rrbracket \implies (\text{*f* exp}) (x::\text{hypreal}) \in \text{HInfinite}$   
 ⟨*proof*⟩

**lemma** *starfun-exp-minus*:  $\forall x. (\text{*f* exp}) (-x) = \text{inverse}((\text{*f* exp}) x)$   
 ⟨*proof*⟩

**lemma** *starfun-exp-Infinitesimal*:  
 $\llbracket x \in \text{HInfinite}; x \leq 0 \rrbracket \implies (\text{*f* exp}) (x::\text{hypreal}) \in \text{Infinitesimal}$   
 ⟨*proof*⟩

**lemma** *starfun-exp-gt-one* [*simp*]:  $\forall x::\text{hypreal}. 0 < x \implies 1 < (\text{*f* exp}) x$   
 ⟨*proof*⟩

**lemma** *starfun-ln-exp* [*simp*]:  $\forall x. (\text{*f* ln}) ((\text{*f* exp}) x) = x$   
 ⟨*proof*⟩

**lemma** *starfun-exp-ln-iff* [*simp*]:  $\forall x. ((\text{*f* exp})((\text{*f* ln}) x) = x) = (0 < x)$   
 ⟨*proof*⟩

**lemma** *starfun-exp-ln-eq*:  $!!u x. (*f* exp) u = x ==> (*f* ln) x = u$   
 ⟨proof⟩

**lemma** *starfun-ln-less-self* [simp]:  $!!x. 0 < x ==> (*f* ln) x < x$   
 ⟨proof⟩

**lemma** *starfun-ln-ge-zero* [simp]:  $!!x. 1 \leq x ==> 0 \leq (*f* ln) x$   
 ⟨proof⟩

**lemma** *starfun-ln-gt-zero* [simp]:  $!!x. 1 < x ==> 0 < (*f* ln) x$   
 ⟨proof⟩

**lemma** *starfun-ln-not-eq-zero* [simp]:  $!!x. [| 0 < x; x \neq 1 |] ==> (*f* ln) x \neq 0$   
 ⟨proof⟩

**lemma** *starfun-ln-HFinite*:  $[| x \in HFinite; 1 \leq x |] ==> (*f* ln) x \in HFinite$   
 ⟨proof⟩

**lemma** *starfun-ln-inverse*:  $!!x. 0 < x ==> (*f* ln) (inverse x) = -( *f* ln) x$   
 ⟨proof⟩

**lemma** *starfun-abs-exp-cancel*:  $\bigwedge x. |( *f* exp) (x::hypreal)| = ( *f* exp) x$   
 ⟨proof⟩

**lemma** *starfun-exp-less-mono*:  $\bigwedge x y::hypreal. x < y ==> (*f* exp) x < (*f* exp) y$   
 ⟨proof⟩

**lemma** *starfun-exp-HFinite*:  $x \in HFinite ==> (*f* exp) (x::hypreal) \in HFinite$   
 ⟨proof⟩

**lemma** *starfun-exp-add-HFinite-Infinitesimal-approx*:  
 $[| x \in Infinitesimal; z \in HFinite |] ==> (*f* exp) (z + x::hypreal) @= (*f* exp) z$   
 ⟨proof⟩

**lemma** *starfun-ln-HInfinite*:  
 $[| x \in HInfinite; 0 < x |] ==> (*f* ln) x \in HInfinite$   
 ⟨proof⟩

**lemma** *starfun-exp-HInfinite-Infinitesimal-disj*:  
 $x \in HInfinite ==> (*f* exp) x \in HInfinite \mid (*f* exp) (x::hypreal) \in Infinitesimal$   
 ⟨proof⟩

**lemma** *starfun-ln-HFinite-not-Infinitesimal*:  
 $[| x \in HFinite - Infinitesimal; 0 < x |] ==> (*f* ln) x \in HFinite$   
 ⟨proof⟩

**lemma** *starfun-ln-Infinitesimal-HInfinite*:

$\llbracket x \in \text{Infinitesimal}; 0 < x \rrbracket \implies (*f* \ln) x \in \text{HInfinite}$   
 $\langle \text{proof} \rangle$

**lemma** *starfun-ln-less-zero*:  $\forall x. \llbracket 0 < x; x < 1 \rrbracket \implies (*f* \ln) x < 0$

$\langle \text{proof} \rangle$

**lemma** *starfun-ln-Infinitesimal-less-zero*:

$\llbracket x \in \text{Infinitesimal}; 0 < x \rrbracket \implies (*f* \ln) x < 0$   
 $\langle \text{proof} \rangle$

**lemma** *starfun-ln-HInfinite-gt-zero*:

$\llbracket x \in \text{HInfinite}; 0 < x \rrbracket \implies 0 < (*f* \ln) x$   
 $\langle \text{proof} \rangle$

**lemma** *HFinite-sin [simp]*:  $\text{sumhr } (0, \text{whn}, \%n. \text{sin-coeff } n * x ^ n) \in \text{HFinite}$

$\langle \text{proof} \rangle$

**lemma** *STAR-sin-zero [simp]*:  $(*f* \sin) 0 = 0$

$\langle \text{proof} \rangle$

**lemma** *STAR-sin-Infinitesimal [simp]*:  $x \in \text{Infinitesimal} \implies (*f* \sin) x @= x$

$\langle \text{proof} \rangle$

**lemma** *HFinite-cos [simp]*:  $\text{sumhr } (0, \text{whn}, \%n. \text{cos-coeff } n * x ^ n) \in \text{HFinite}$

$\langle \text{proof} \rangle$

**lemma** *STAR-cos-zero [simp]*:  $(*f* \cos) 0 = 1$

$\langle \text{proof} \rangle$

**lemma** *STAR-cos-Infinitesimal [simp]*:  $x \in \text{Infinitesimal} \implies (*f* \cos) x @= 1$

$\langle \text{proof} \rangle$

**lemma** *STAR-tan-zero [simp]*:  $(*f* \tan) 0 = 0$

$\langle \text{proof} \rangle$

**lemma** *STAR-tan-Infinitesimal*:  $x \in \text{Infinitesimal} \implies (*f* \tan) x @= x$

$\langle \text{proof} \rangle$

**lemma** *STAR-sin-cos-Infinitesimal-mult*:

$x \in \text{Infinitesimal} \implies (*f* \sin) x * (*f* \cos) x @= x$   
 $\langle \text{proof} \rangle$

**lemma** *HFinite-pi*:  $\text{hypreal-of-real } \pi \in \text{HFinite}$

$\langle proof \rangle$

**lemma** *lemma-split-hypreal-of-real*:

$N \in \text{HNatInfinite}$

$\implies \text{hypreal-of-real } a =$

$\text{hypreal-of-hypnat } N * (\text{inverse}(\text{hypreal-of-hypnat } N) * \text{hypreal-of-real } a)$

$\langle proof \rangle$

**lemma** *STAR-sin-Infinitesimal-divide*:

$[[x \in \text{Infinitesimal}; x \neq 0]] \implies (*f* \sin) x/x @= 1$

$\langle proof \rangle$

**lemma** *lemma-sin-pi*:

$n \in \text{HNatInfinite}$

$\implies (*f* \sin) (\text{inverse}(\text{hypreal-of-hypnat } n)) / (\text{inverse}(\text{hypreal-of-hypnat } n)) @= 1$

$\langle proof \rangle$

**lemma** *STAR-sin-inverse-HNatInfinite*:

$n \in \text{HNatInfinite}$

$\implies (*f* \sin) (\text{inverse}(\text{hypreal-of-hypnat } n)) * \text{hypreal-of-hypnat } n @= 1$

$\langle proof \rangle$

**lemma** *Infinitesimal-pi-divide-HNatInfinite*:

$N \in \text{HNatInfinite}$

$\implies \text{hypreal-of-real } \pi / (\text{hypreal-of-hypnat } N) \in \text{Infinitesimal}$

$\langle proof \rangle$

**lemma** *pi-divide-HNatInfinite-not-zero [simp]*:

$N \in \text{HNatInfinite} \implies \text{hypreal-of-real } \pi / (\text{hypreal-of-hypnat } N) \neq 0$

$\langle proof \rangle$

**lemma** *STAR-sin-pi-divide-HNatInfinite-approx-pi*:

$n \in \text{HNatInfinite}$

$\implies (*f* \sin) (\text{hypreal-of-real } \pi / (\text{hypreal-of-hypnat } n)) * \text{hypreal-of-hypnat } n$

$n$

$@= \text{hypreal-of-real } \pi$

$\langle proof \rangle$

**lemma** *STAR-sin-pi-divide-HNatInfinite-approx-pi2*:

$n \in \text{HNatInfinite}$

$\implies \text{hypreal-of-hypnat } n *$

$(*f* \sin) (\text{hypreal-of-real } \pi / (\text{hypreal-of-hypnat } n))$

$\text{@} = \text{hypreal-of-real } \pi$   
 ⟨proof⟩

**lemma** *starfunNat-pi-divide-n-Infinitesimal*:

$N \in \text{HNatInfinite} \implies (*f* (\%x. \pi / \text{real } x)) N \in \text{Infinitesimal}$   
 ⟨proof⟩

**lemma** *STAR-sin-pi-divide-n-approx*:

$N \in \text{HNatInfinite} \implies$   
 $( *f* \text{ sin } ) ( ( *f* (\%x. \pi / \text{real } x) ) N ) \text{@} =$   
 $\text{hypreal-of-real } \pi / (\text{hypreal-of-hypnat } N)$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-sin-pi*:  $(\%n. \text{real } n * \text{ sin } (\pi / \text{real } n)) \text{-----NS} > \pi$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-cos-one*:  $(\%n. \text{ cos } (\pi / \text{real } n)) \text{-----NS} > 1$   
 ⟨proof⟩

**lemma** *NSLIMSEQ-sin-cos-pi*:

$(\%n. \text{real } n * \text{ sin } (\pi / \text{real } n) * \text{ cos } (\pi / \text{real } n)) \text{-----NS} > \pi$   
 ⟨proof⟩

A familiar approximation to  $\cos x$  when  $x$  is small

**lemma** *STAR-cos-Infinitesimal-approx*:

$x \in \text{Infinitesimal} \implies (*f* \text{ cos } ) x \text{@} = 1 - x ^ 2$   
 ⟨proof⟩

**lemma** *STAR-cos-Infinitesimal-approx2*:

$x \in \text{Infinitesimal} \implies (*f* \text{ cos } ) x \text{@} = 1 - (x ^ 2)/2$   
 ⟨proof⟩

**end**

## 17 NSCA: Non-Standard Complex Analysis

**theory** *NSCA*

**imports** *NSComplex HTranscendental*

**begin**

**abbreviation**

$\text{SComplex} :: \text{hcomplex set}$  **where**  
 $\text{SComplex} \equiv \text{Standard}$

**definition** — standard part map

$\text{stc} :: \text{hcomplex} \Rightarrow \text{hcomplex}$  **where**  
 $\text{stc } x = (\text{SOME } r. x \in \text{HFinite} \ \& \ r : \text{SComplex} \ \& \ r \text{@} = x)$

## 17.1 Closure Laws for SComplex, the Standard Complex Numbers

**lemma** *SComplex-minus-iff* [simp]:  $(-x \in SComplex) = (x \in SComplex)$   
 ⟨proof⟩

**lemma** *SComplex-add-cancel*:  
 $[[ x + y \in SComplex; y \in SComplex ]] ==> x \in SComplex$   
 ⟨proof⟩

**lemma** *SReal-hcmod-hcomplex-of-complex* [simp]:  
 $hcmod (hcomplex-of-complex r) \in Reals$   
 ⟨proof⟩

**lemma** *SReal-hcmod-numeral* [simp]:  $hcmod (numeral w :: hcomplex) \in Reals$   
 ⟨proof⟩

**lemma** *SReal-hcmod-SComplex*:  $x \in SComplex ==> hcmod x \in Reals$   
 ⟨proof⟩

**lemma** *SComplex-divide-numeral*:  
 $r \in SComplex ==> r / (numeral w :: hcomplex) \in SComplex$   
 ⟨proof⟩

**lemma** *SComplex-UNIV-complex*:  
 $\{x. hcomplex-of-complex x \in SComplex\} = (UNIV :: complex set)$   
 ⟨proof⟩

**lemma** *SComplex-iff*:  $(x \in SComplex) = (\exists y. x = hcomplex-of-complex y)$   
 ⟨proof⟩

**lemma** *hcomplex-of-complex-image*:  
 $hcomplex-of-complex `(UNIV :: complex set) = SComplex$   
 ⟨proof⟩

**lemma** *inv-hcomplex-of-complex-image*:  $inv hcomplex-of-complex `SComplex = UNIV$   
 ⟨proof⟩

**lemma** *SComplex-hcomplex-of-complex-image*:  
 $[[ \exists x. x: P; P \leq SComplex ]] ==> \exists Q. P = hcomplex-of-complex ` Q$   
 ⟨proof⟩

**lemma** *SComplex-SReal-dense*:  
 $[[ x \in SComplex; y \in SComplex; hcmod x < hcmod y ]]$   
 $==> \exists r \in Reals. hcmod x < r \ \& \ r < hcmod y$   
 ⟨proof⟩

**lemma** *SComplex-hcmod-SReal*:  
 $z \in SComplex ==> hcmod z \in Reals$   
 ⟨proof⟩

## 17.2 The Finite Elements form a Subring

**lemma** *HFinite-hcmod-hcomplex-of-complex* [simp]:

$$\text{hcmod } (\text{hcomplex-of-complex } r) \in \text{HFinite}$$

*<proof>*

**lemma** *HFinite-hcmod-iff*:  $(x \in \text{HFinite}) = (\text{hcmod } x \in \text{HFinite})$

*<proof>*

**lemma** *HFinite-bounded-hcmod*:

$$[\![ x \in \text{HFinite}; y \leq \text{hcmod } x; 0 \leq y \!\!] \implies y \in \text{HFinite}$$

*<proof>*

## 17.3 The Complex Infinitesimals form a Subring

**lemma** *hcomplex-sum-of-halves*:  $x/(2::\text{hcomplex}) + x/(2::\text{hcomplex}) = x$

*<proof>*

**lemma** *Infinitesimal-hcmod-iff*:

$$(z \in \text{Infinitesimal}) = (\text{hcmod } z \in \text{Infinitesimal})$$

*<proof>*

**lemma** *HInfinite-hcmod-iff*:  $(z \in \text{HInfinite}) = (\text{hcmod } z \in \text{HInfinite})$

*<proof>*

**lemma** *HFinite-diff-Infinitesimal-hcmod*:

$$x \in \text{HFinite} - \text{Infinitesimal} \implies \text{hcmod } x \in \text{HFinite} - \text{Infinitesimal}$$

*<proof>*

**lemma** *hcmod-less-Infinitesimal*:

$$[\![ e \in \text{Infinitesimal}; \text{hcmod } x < \text{hcmod } e \!\!] \implies x \in \text{Infinitesimal}$$

*<proof>*

**lemma** *hcmod-le-Infinitesimal*:

$$[\![ e \in \text{Infinitesimal}; \text{hcmod } x \leq \text{hcmod } e \!\!] \implies x \in \text{Infinitesimal}$$

*<proof>*

**lemma** *Infinitesimal-interval-hcmod*:

$$\begin{aligned} &[\![ e \in \text{Infinitesimal}; \\ &\quad e' \in \text{Infinitesimal}; \\ &\quad \text{hcmod } e' < \text{hcmod } x ; \text{hcmod } x < \text{hcmod } e \\ &\!\!] \implies x \in \text{Infinitesimal} \end{aligned}$$

*<proof>*

**lemma** *Infinitesimal-interval2-hcmod*:

$$\begin{aligned} &[\![ e \in \text{Infinitesimal}; \\ &\quad e' \in \text{Infinitesimal}; \\ &\quad \text{hcmod } e' \leq \text{hcmod } x ; \text{hcmod } x \leq \text{hcmod } e \\ &\!\!] \implies x \in \text{Infinitesimal} \end{aligned}$$

*<proof>*

## 17.4 The “Infinitely Close” Relation

**lemma** *approx-SComplex-mult-cancel-zero*:

$\llbracket a \in SComplex; a \neq 0; a*x \text{ @} = 0 \rrbracket \implies x \text{ @} = 0$   
*<proof>*

**lemma** *approx-mult-SComplex1*:  $\llbracket a \in SComplex; x \text{ @} = 0 \rrbracket \implies x*a \text{ @} = 0$

*<proof>*

**lemma** *approx-mult-SComplex2*:  $\llbracket a \in SComplex; x \text{ @} = 0 \rrbracket \implies a*x \text{ @} = 0$

*<proof>*

**lemma** *approx-mult-SComplex-zero-cancel-iff* [simp]:

$\llbracket a \in SComplex; a \neq 0 \rrbracket \implies (a*x \text{ @} = 0) = (x \text{ @} = 0)$   
*<proof>*

**lemma** *approx-SComplex-mult-cancel*:

$\llbracket a \in SComplex; a \neq 0; a*w \text{ @} = a*z \rrbracket \implies w \text{ @} = z$   
*<proof>*

**lemma** *approx-SComplex-mult-cancel-iff1* [simp]:

$\llbracket a \in SComplex; a \neq 0 \rrbracket \implies (a*w \text{ @} = a*z) = (w \text{ @} = z)$   
*<proof>*

**lemma** *approx-hcmod-approx-zero*:  $(x \text{ @} = y) = (hcmod (y - x) \text{ @} = 0)$

*<proof>*

**lemma** *approx-approx-zero-iff*:  $(x \text{ @} = 0) = (hcmod x \text{ @} = 0)$

*<proof>*

**lemma** *approx-minus-zero-cancel-iff* [simp]:  $(-x \text{ @} = 0) = (x \text{ @} = 0)$

*<proof>*

**lemma** *Infinitesimal-hcmod-add-diff*:

$u \text{ @} = 0 \implies hcmod(x + u) - hcmod x \in Infinitesimal$   
*<proof>*

**lemma** *approx-hcmod-add-hcmod*:  $u \text{ @} = 0 \implies hcmod(x + u) \text{ @} = hcmod x$

*<proof>*

## 17.5 Zero is the Only Infinitesimal Complex Number

**lemma** *Infinitesimal-less-SComplex*:

$\llbracket x \in SComplex; y \in Infinitesimal; 0 < hcmod x \rrbracket \implies hcmod y < hcmod x$   
*<proof>*

**lemma** *SComplex-Int-Infinitesimal-zero*:  $SComplex \text{ Int } Infinitesimal = \{0\}$

*<proof>*

**lemma** *SComplex-Infinitesimal-zero*:

$[[ x \in SComplex; x \in Infinitesimal ]] ==> x = 0$   
 ⟨proof⟩

**lemma** *SComplex-HFinite-diff-Infinitesimal*:

$[[ x \in SComplex; x \neq 0 ]] ==> x \in HFinite - Infinitesimal$   
 ⟨proof⟩

**lemma** *hcomplex-of-complex-HFinite-diff-Infinitesimal*:

*hcomplex-of-complex*  $x \neq 0$   
 $==> hcomplex-of-complex x \in HFinite - Infinitesimal$   
 ⟨proof⟩

**lemma** *numeral-not-Infinitesimal [simp]*:

*numeral*  $w \neq (0::hcomplex) ==> (numeral w::hcomplex) \notin Infinitesimal$   
 ⟨proof⟩

**lemma** *approx-SComplex-not-zero*:

$[[ y \in SComplex; x @= y; y \neq 0 ]] ==> x \neq 0$   
 ⟨proof⟩

**lemma** *SComplex-approx-iff*:

$[[ x \in SComplex; y \in SComplex ]] ==> (x @= y) = (x = y)$   
 ⟨proof⟩

**lemma** *numeral-Infinitesimal-iff [simp]*:

$((numeral w :: hcomplex) \in Infinitesimal) =$   
 $(numeral w = (0::hcomplex))$   
 ⟨proof⟩

**lemma** *approx-unique-complex*:

$[[ r \in SComplex; s \in SComplex; r @= x; s @= x ]] ==> r = s$   
 ⟨proof⟩

## 17.6 Properties of *hRe*, *hIm* and *HComplex*

**lemma** *abs-hRe-le-hcmod*:  $\bigwedge x. |hRe x| \leq hcmod x$

⟨proof⟩

**lemma** *abs-hIm-le-hcmod*:  $\bigwedge x. |hIm x| \leq hcmod x$

⟨proof⟩

**lemma** *Infinitesimal-hRe*:  $x \in Infinitesimal \implies hRe x \in Infinitesimal$

⟨proof⟩

**lemma** *Infinitesimal-hIm*:  $x \in Infinitesimal \implies hIm x \in Infinitesimal$

⟨proof⟩

**lemma** *real-sqrt-lessI*:  $\llbracket 0 < u; x < u^2 \rrbracket \implies \text{sqrt } x < u$

*<proof>*

**lemma** *hypreal-sqrt-lessI*:

$\bigwedge x u. \llbracket 0 < u; x < u^2 \rrbracket \implies (*f* \text{ sqrt}) x < u$   
*<proof>*

**lemma** *hypreal-sqrt-ge-zero*:  $\bigwedge x. 0 \leq x \implies 0 \leq (*f* \text{ sqrt}) x$

*<proof>*

**lemma** *Infinitesimal-sqrt*:

$\llbracket x \in \text{Infinitesimal}; 0 \leq x \rrbracket \implies (*f* \text{ sqrt}) x \in \text{Infinitesimal}$   
*<proof>*

**lemma** *Infinitesimal-HComplex*:

$\llbracket x \in \text{Infinitesimal}; y \in \text{Infinitesimal} \rrbracket \implies \text{HComplex } x y \in \text{Infinitesimal}$   
*<proof>*

**lemma** *hcomplex-Infinitesimal-iff*:

$(x \in \text{Infinitesimal}) = (\text{hRe } x \in \text{Infinitesimal} \wedge \text{hIm } x \in \text{Infinitesimal})$   
*<proof>*

**lemma** *hRe-diff [simp]*:  $\bigwedge x y. \text{hRe } (x - y) = \text{hRe } x - \text{hRe } y$

*<proof>*

**lemma** *hIm-diff [simp]*:  $\bigwedge x y. \text{hIm } (x - y) = \text{hIm } x - \text{hIm } y$

*<proof>*

**lemma** *approx-hRe*:  $x \approx y \implies \text{hRe } x \approx \text{hRe } y$

*<proof>*

**lemma** *approx-hIm*:  $x \approx y \implies \text{hIm } x \approx \text{hIm } y$

*<proof>*

**lemma** *approx-HComplex*:

$\llbracket a \approx b; c \approx d \rrbracket \implies \text{HComplex } a c \approx \text{HComplex } b d$   
*<proof>*

**lemma** *hcomplex-approx-iff*:

$(x \approx y) = (\text{hRe } x \approx \text{hRe } y \wedge \text{hIm } x \approx \text{hIm } y)$   
*<proof>*

**lemma** *HFinite-hRe*:  $x \in \text{HFinite} \implies \text{hRe } x \in \text{HFinite}$

*<proof>*

**lemma** *HFinite-hIm*:  $x \in \text{HFinite} \implies \text{hIm } x \in \text{HFinite}$

*<proof>*

**lemma** *HFinite-HComplex*:

$\llbracket x \in HFinite; y \in HFinite \rrbracket \implies HComplex\ x\ y \in HFinite$   
 ⟨proof⟩

**lemma** *hcomplex-HFinite-iff*:

$(x \in HFinite) = (hRe\ x \in HFinite \wedge hIm\ x \in HFinite)$   
 ⟨proof⟩

**lemma** *hcomplex-HInfinite-iff*:

$(x \in HInfinite) = (hRe\ x \in HInfinite \vee hIm\ x \in HInfinite)$   
 ⟨proof⟩

**lemma** *hcomplex-of-hypreal-approx-iff [simp]*:

$(hcomplex-of-hypreal\ x\ @= hcomplex-of-hypreal\ z) = (x\ @= z)$   
 ⟨proof⟩

**lemma** *Standard-HComplex*:

$\llbracket x \in Standard; y \in Standard \rrbracket \implies HComplex\ x\ y \in Standard$   
 ⟨proof⟩

**lemma** *stc-part-Ex*:  $x:HFinite \implies \exists t \in SComplex. x\ @= t$

⟨proof⟩

**lemma** *stc-part-Ex1*:  $x:HFinite \implies EX! t. t \in SComplex \ \& \ x\ @= t$

⟨proof⟩

**lemmas** *hcomplex-of-complex-approx-inverse =*

*hcomplex-of-complex-HFinite-diff-Infinitesimal [THEN [2] approx-inverse]*

## 17.7 Theorems About Monads

**lemma** *monad-zero-hcmod-iff*:  $(x \in monad\ 0) = (hcmod\ x:monad\ 0)$

⟨proof⟩

## 17.8 Theorems About Standard Part

**lemma** *stc-approx-self*:  $x \in HFinite \implies stc\ x\ @= x$

⟨proof⟩

**lemma** *stc-SComplex*:  $x \in HFinite \implies stc\ x \in SComplex$

⟨proof⟩

**lemma** *stc-HFinite*:  $x \in HFinite \implies stc\ x \in HFinite$

⟨proof⟩

**lemma** *stc-unique*:  $\llbracket y \in SComplex; y \approx x \rrbracket \implies stc\ x = y$

⟨proof⟩

**lemma** *stc-SComplex-eq [simp]*:  $x \in SComplex \implies stc\ x = x$

*<proof>*

**lemma** *stc-hcomplex-of-complex*:

$$\text{stc} (\text{hcomplex-of-complex } x) = \text{hcomplex-of-complex } x$$

*<proof>*

**lemma** *stc-eq-approx*:

$$[[ x \in \text{HFinite}; y \in \text{HFinite}; \text{stc } x = \text{stc } y ]] ==> x @= y$$

*<proof>*

**lemma** *approx-stc-eq*:

$$[[ x \in \text{HFinite}; y \in \text{HFinite}; x @= y ]] ==> \text{stc } x = \text{stc } y$$

*<proof>*

**lemma** *stc-eq-approx-iff*:

$$[[ x \in \text{HFinite}; y \in \text{HFinite} ]] ==> (x @= y) = (\text{stc } x = \text{stc } y)$$

*<proof>*

**lemma** *stc-Infinitesimal-add-SComplex*:

$$[[ x \in \text{SComplex}; e \in \text{Infinitesimal} ]] ==> \text{stc}(x + e) = x$$

*<proof>*

**lemma** *stc-Infinitesimal-add-SComplex2*:

$$[[ x \in \text{SComplex}; e \in \text{Infinitesimal} ]] ==> \text{stc}(e + x) = x$$

*<proof>*

**lemma** *HFinite-stc-Infinitesimal-add*:

$$x \in \text{HFinite} ==> \exists e \in \text{Infinitesimal}. x = \text{stc}(x) + e$$

*<proof>*

**lemma** *stc-add*:

$$[[ x \in \text{HFinite}; y \in \text{HFinite} ]] ==> \text{stc}(x + y) = \text{stc}(x) + \text{stc}(y)$$

*<proof>*

**lemma** *stc-numeral [simp]*:  $\text{stc} (\text{numeral } w) = \text{numeral } w$

*<proof>*

**lemma** *stc-zero [simp]*:  $\text{stc } 0 = 0$

*<proof>*

**lemma** *stc-one [simp]*:  $\text{stc } 1 = 1$

*<proof>*

**lemma** *stc-minus*:  $y \in \text{HFinite} ==> \text{stc}(-y) = -\text{stc}(y)$

*<proof>*

**lemma** *stc-diff*:

$$[[ x \in \text{HFinite}; y \in \text{HFinite} ]] ==> \text{stc}(x - y) = \text{stc}(x) - \text{stc}(y)$$

*<proof>*

**lemma** *stc-mult*:

$$\begin{aligned} & [| x \in \mathit{HFinite}; y \in \mathit{HFinite} |] \\ & \implies \mathit{stc} (x * y) = \mathit{stc}(x) * \mathit{stc}(y) \end{aligned}$$

*<proof>*

**lemma** *stc-Infinitesimal*:  $x \in \mathit{Infinitesimal} \implies \mathit{stc} x = 0$

*<proof>*

**lemma** *stc-not-Infinitesimal*:  $\mathit{stc}(x) \neq 0 \implies x \notin \mathit{Infinitesimal}$

*<proof>*

**lemma** *stc-inverse*:

$$\begin{aligned} & [| x \in \mathit{HFinite}; \mathit{stc} x \neq 0 |] \\ & \implies \mathit{stc}(\mathit{inverse} x) = \mathit{inverse} (\mathit{stc} x) \end{aligned}$$

*<proof>*

**lemma** *stc-divide* [*simp*]:

$$\begin{aligned} & [| x \in \mathit{HFinite}; y \in \mathit{HFinite}; \mathit{stc} y \neq 0 |] \\ & \implies \mathit{stc}(x/y) = (\mathit{stc} x) / (\mathit{stc} y) \end{aligned}$$

*<proof>*

**lemma** *stc-idempotent* [*simp*]:  $x \in \mathit{HFinite} \implies \mathit{stc}(\mathit{stc}(x)) = \mathit{stc}(x)$

*<proof>*

**lemma** *HFinite-HFinite-hcomplex-of-hypreal*:

$$z \in \mathit{HFinite} \implies \mathit{hcomplex-of-hypreal} z \in \mathit{HFinite}$$

*<proof>*

**lemma** *SComplex-SReal-hcomplex-of-hypreal*:

$$x \in \mathit{Reals} \implies \mathit{hcomplex-of-hypreal} x \in \mathit{SComplex}$$

*<proof>*

**lemma** *stc-hcomplex-of-hypreal*:

$$z \in \mathit{HFinite} \implies \mathit{stc}(\mathit{hcomplex-of-hypreal} z) = \mathit{hcomplex-of-hypreal} (\mathit{st} z)$$

*<proof>*

**lemma** *Infinitesimal-hcnj-iff* [*simp*]:

$$(\mathit{hcnj} z \in \mathit{Infinitesimal}) = (z \in \mathit{Infinitesimal})$$

*<proof>*

**lemma** *Infinitesimal-hcomplex-of-hypreal-epsilon* [*simp*]:

$$\mathit{hcomplex-of-hypreal} \mathit{epsilon} \in \mathit{Infinitesimal}$$

*<proof>*

**end**

## 18 CStar: Star-transforms in NSA, Extending Sets of Complex Numbers and Complex Functions

```
theory CStar
imports NSCA
begin
```

### 18.1 Properties of the \*-Transform Applied to Sets of Reals

**lemma** *STARC-hcomplex-of-complex-Int*:

```
  ** X Int SComplex = hcomplex-of-complex ‘ X
⟨proof⟩
```

**lemma** *lemma-not-hcomplexA*:

```
  x ∉ hcomplex-of-complex ‘ A ==> ∀ y ∈ A. x ≠ hcomplex-of-complex y
⟨proof⟩
```

### 18.2 Theorems about Nonstandard Extensions of Functions

**lemma** *starfunC-hcpow*:  $!!Z. (*f* (%z. z ^ n)) Z = Z \text{ pow hypnat-of-nat } n$   
 ⟨proof⟩

**lemma** *starfunCR-cmod*:  $*f* \text{ cmod} = \text{hcm}od$   
 ⟨proof⟩

### 18.3 Internal Functions - Some Redundancy With \*f\* Now

**lemma** *starfun-Re*:  $(*f* (\lambda x. \text{Re} (f x))) = (\lambda x. \text{hRe} (( *f* f) x))$   
 ⟨proof⟩

**lemma** *starfun-Im*:  $(*f* (\lambda x. \text{Im} (f x))) = (\lambda x. \text{hIm} (( *f* f) x))$   
 ⟨proof⟩

**lemma** *starfunC-eq-Re-Im-iff*:

```
  (( *f* f) x = z) = ((( *f* (%x. \text{Re}(f x))) x = \text{hRe} (z)) &
    (( *f* (%x. \text{Im}(f x))) x = \text{hIm} (z)))
⟨proof⟩
```

**lemma** *starfunC-approx-Re-Im-iff*:

```
  (( *f* f) x @= z) = ((( *f* (%x. \text{Re}(f x))) x @= \text{hRe} (z)) &
    (( *f* (%x. \text{Im}(f x))) x @= \text{hIm} (z)))
⟨proof⟩
```

```
end
```

## 19 CLim: Limits, Continuity and Differentiation for Complex Functions

```
theory CLim
imports CStar
begin
```

```
declare hypreal-epsilon-not-zero [simp]
```

```
lemma lemma-complex-mult-inverse-squared [simp]:
   $x \neq (0::\text{complex}) \implies (x * \text{inverse}(x) ^ 2) = \text{inverse } x$ 
  <proof>
```

Changing the quantified variable. Install earlier?

```
lemma all-shift:  $(\forall x::'a::\text{comm-ring-1}. P x) = (\forall x. P (x-a))$ 
  <proof>
```

```
lemma complex-add-minus-iff [simp]:  $(x + - a = (0::\text{complex})) = (x=a)$ 
  <proof>
```

```
lemma complex-add-eq-0-iff [iff]:  $(x+y = (0::\text{complex})) = (y = -x)$ 
  <proof>
```

### 19.1 Limit of Complex to Complex Function

```
lemma NSLIM-Re:  $f \dashrightarrow a \dashrightarrow NS > L \implies (\%x. \text{Re}(f x)) \dashrightarrow a \dashrightarrow NS > \text{Re}(L)$ 
  <proof>
```

```
lemma NSLIM-Im:  $f \dashrightarrow a \dashrightarrow NS > L \implies (\%x. \text{Im}(f x)) \dashrightarrow a \dashrightarrow NS > \text{Im}(L)$ 
  <proof>
```

```
lemma LIM-Re:
  fixes  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$ 
  shows  $f \dashrightarrow a \dashrightarrow L \implies (\%x. \text{Re}(f x)) \dashrightarrow a \dashrightarrow \text{Re}(L)$ 
  <proof>
```

```
lemma LIM-Im:
  fixes  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$ 
  shows  $f \dashrightarrow a \dashrightarrow L \implies (\%x. \text{Im}(f x)) \dashrightarrow a \dashrightarrow \text{Im}(L)$ 
  <proof>
```

```
lemma LIM-cnj:
  fixes  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$ 
  shows  $f \dashrightarrow a \dashrightarrow L \implies (\%x. \text{cnj}(f x)) \dashrightarrow a \dashrightarrow \text{cnj } L$ 
  <proof>
```

**lemma** *LIM-cnj-iff*:

**fixes**  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$

**shows**  $((\%x. \text{cnj } (f x)) \dashv\vdash a \dashv\vdash \text{cnj } L) = (f \dashv\vdash a \dashv\vdash L)$   
 $\langle \text{proof} \rangle$

**lemma** *starfun-norm*:  $( *f* (\lambda x. \text{norm } (f x))) = (\lambda x. \text{hnorm } (( *f* f) x))$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-Re [simp]*:  $\text{star-of } (\text{Re } x) = \text{hRe } (\text{star-of } x)$   
 $\langle \text{proof} \rangle$

**lemma** *star-of-Im [simp]*:  $\text{star-of } (\text{Im } x) = \text{hIm } (\text{star-of } x)$   
 $\langle \text{proof} \rangle$

**lemma** *NSCLIM-NSCRLIM-iff*:

$(f \dashv\vdash x \dashv\vdash \text{NS} > L) = ((\%y. \text{cmod}(f y - L)) \dashv\vdash x \dashv\vdash \text{NS} > 0)$   
 $\langle \text{proof} \rangle$

**lemma** *CLIM-CRLIM-iff*:

**fixes**  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$

**shows**  $(f \dashv\vdash x \dashv\vdash L) = ((\%y. \text{cmod}(f y - L)) \dashv\vdash x \dashv\vdash 0)$   
 $\langle \text{proof} \rangle$

**lemma** *NSCLIM-NSCRLIM-iff2*:

$(f \dashv\vdash x \dashv\vdash \text{NS} > L) = ((\%y. \text{cmod}(f y - L)) \dashv\vdash x \dashv\vdash \text{NS} > 0)$   
 $\langle \text{proof} \rangle$

**lemma** *NSLIM-NSCRLIM-Re-Im-iff*:

$(f \dashv\vdash a \dashv\vdash \text{NS} > L) = ((\%x. \text{Re}(f x)) \dashv\vdash a \dashv\vdash \text{NS} > \text{Re}(L) \ \&$   
 $(\%x. \text{Im}(f x)) \dashv\vdash a \dashv\vdash \text{NS} > \text{Im}(L))$

$\langle \text{proof} \rangle$

**lemma** *LIM-CRLIM-Re-Im-iff*:

**fixes**  $f :: 'a::\text{real-normed-vector} \Rightarrow \text{complex}$

**shows**  $(f \dashv\vdash a \dashv\vdash L) = ((\%x. \text{Re}(f x)) \dashv\vdash a \dashv\vdash \text{Re}(L) \ \&$   
 $(\%x. \text{Im}(f x)) \dashv\vdash a \dashv\vdash \text{Im}(L))$

$\langle \text{proof} \rangle$

## 19.2 Continuity

**lemma** *NSLIM-isContc-iff*:

$(f \dashv\vdash a \dashv\vdash \text{NS} > f a) = ((\%h. f(a + h)) \dashv\vdash 0 \dashv\vdash \text{NS} > f a)$   
 $\langle \text{proof} \rangle$

## 19.3 Functions from Complex to Reals

**lemma** *isNSContCR-cmod [simp]*:  $\text{isNSCont } \text{cmod } (a)$

$\langle \text{proof} \rangle$

**lemma** *isContCR-cmod* [simp]: *isCont cmod* (*a*)  
 ⟨proof⟩

**lemma** *isCont-Re*:  
 fixes *f* :: '*a*::real-normed-vector ⇒ complex  
 shows *isCont f a* ==> *isCont* (%*x*. *Re* (*f x*)) *a*  
 ⟨proof⟩

**lemma** *isCont-Im*:  
 fixes *f* :: '*a*::real-normed-vector ⇒ complex  
 shows *isCont f a* ==> *isCont* (%*x*. *Im* (*f x*)) *a*  
 ⟨proof⟩

## 19.4 Differentiation of Natural Number Powers

**lemma** *CDERIV-pow* [simp]:  
 $DERIV$  (%*x*.  $x^n$ ) *x* := (*complex-of-real* (*real n*)) \* ( $x^{n - Suc\ 0}$ )  
 ⟨proof⟩

Nonstandard version

**lemma** *NSCDERIV-pow*:  
 $NSDERIV$  (%*x*.  $x^n$ ) *x* := *complex-of-real* (*real n*) \* ( $x^{n - 1}$ )  
 ⟨proof⟩

Can't relax the premise  $x \neq (0::'a)$ : it isn't continuous at zero

**lemma** *NSCDERIV-inverse*:  
 $(x::complex) \neq 0$  ==>  $NSDERIV$  (%*x*. *inverse*(*x*)) *x* := ( $-(inverse\ x^2)$ )  
 ⟨proof⟩

**lemma** *CDERIV-inverse*:  
 $(x::complex) \neq 0$  ==>  $DERIV$  (%*x*. *inverse*(*x*)) *x* := ( $-(inverse\ x^2)$ )  
 ⟨proof⟩

## 19.5 Derivative of Reciprocals (Function *inverse*)

**lemma** *CDERIV-inverse-fun*:  
 [|  $DERIV\ f\ x$  := *d*;  $f(x) \neq (0::complex)$  |]  
 ==>  $DERIV$  (%*x*. *inverse*(*f x*)) *x* := ( $-(d * inverse(f(x)^2))$ )  
 ⟨proof⟩

**lemma** *NSCDERIV-inverse-fun*:  
 [|  $NSDERIV\ f\ x$  := *d*;  $f(x) \neq (0::complex)$  |]  
 ==>  $NSDERIV$  (%*x*. *inverse*(*f x*)) *x* := ( $-(d * inverse(f(x)^2))$ )  
 ⟨proof⟩

## 19.6 Derivative of Quotient

**lemma** *CDERIV-quotient*:

```

    [| DERIV f x :> d; DERIV g x :> e; g(x) ≠ (0::complex) |]
      ==> DERIV (%y. f(y) / (g y)) x :> (d*g(x) - (e*f(x))) / (g(x) ^ 2)
  <proof>

```

**lemma** NSCDERIV-quotient:

```

    [| NSDERIV f x :> d; NSDERIV g x :> e; g(x) ≠ (0::complex) |]
      ==> NSDERIV (%y. f(y) / (g y)) x :> (d*g(x) - (e*f(x))) / (g(x) ^ 2)
  <proof>

```

## 19.7 Caratheodory Formulation of Derivative at a Point: Standard Proof

**lemma** CARAT-CDERIVD:

```

    (∀ z. f z - f x = g z * (z - x)) & isNSCont g x & g x = l
      ==> NSDERIV f x :> l
  <proof>

```

**end**

## 20 HLog: Logarithms: Non-Standard Version

```

theory HLog
imports Log HTranscendental
begin

```

**lemma** epsilon-ge-zero [simp]:  $0 \leq \text{epsilon}$   
 <proof>

**lemma** hpfinite-witness:  $\text{epsilon} : \{x. 0 \leq x \ \& \ x : HFinite\}$   
 <proof>

**definition**

```

  powhr :: [hypreal, hypreal] => hypreal   (infixr powhr 80) where
    [transfer-unfold]: x powhr a = starfun2 (op powr) x a

```

**definition**

```

  hlog :: [hypreal, hypreal] => hypreal where
    [transfer-unfold]: hlog a x = starfun2 log a x

```

**lemma** powhr:  $(\text{star-}n \ X) \ \text{powhr} \ (\text{star-}n \ Y) = \text{star-}n \ (\%n. (X \ n) \ \text{powr} \ (Y \ n))$   
 <proof>

**lemma** powhr-one-eq-one [simp]:  $!!a. 1 \ \text{powhr} \ a = 1$   
 <proof>

**lemma** *powhr-mult*:

$!!a\ x\ y.\ [0 < x; 0 < y] \implies (x * y)\ \text{powhr}\ a = (x\ \text{powhr}\ a) * (y\ \text{powhr}\ a)$   
 $\langle\text{proof}\rangle$

**lemma** *powhr-gt-zero* [*simp*]:  $!!a\ x.\ 0 < x\ \text{powhr}\ a$

$\langle\text{proof}\rangle$

**lemma** *powhr-not-zero* [*simp*]:  $x\ \text{powhr}\ a \neq 0$

$\langle\text{proof}\rangle$

**lemma** *powhr-divide*:

$!!a\ x\ y.\ [0 < x; 0 < y] \implies (x / y)\ \text{powhr}\ a = (x\ \text{powhr}\ a) / (y\ \text{powhr}\ a)$   
 $\langle\text{proof}\rangle$

**lemma** *powhr-add*:  $!!a\ b\ x.\ x\ \text{powhr}\ (a + b) = (x\ \text{powhr}\ a) * (x\ \text{powhr}\ b)$

$\langle\text{proof}\rangle$

**lemma** *powhr-powhr*:  $!!a\ b\ x.\ (x\ \text{powhr}\ a)\ \text{powhr}\ b = x\ \text{powhr}\ (a * b)$

$\langle\text{proof}\rangle$

**lemma** *powhr-powhr-swap*:  $!!a\ b\ x.\ (x\ \text{powhr}\ a)\ \text{powhr}\ b = (x\ \text{powhr}\ b)\ \text{powhr}\ a$

$\langle\text{proof}\rangle$

**lemma** *powhr-minus*:  $!!a\ x.\ x\ \text{powhr}\ (-a) = \text{inverse}\ (x\ \text{powhr}\ a)$

$\langle\text{proof}\rangle$

**lemma** *powhr-minus-divide*:  $x\ \text{powhr}\ (-a) = 1 / (x\ \text{powhr}\ a)$

$\langle\text{proof}\rangle$

**lemma** *powhr-less-mono*:  $!!a\ b\ x.\ [a < b; 1 < x] \implies x\ \text{powhr}\ a < x\ \text{powhr}\ b$

$\langle\text{proof}\rangle$

**lemma** *powhr-less-cancel*:  $!!a\ b\ x.\ [x\ \text{powhr}\ a < x\ \text{powhr}\ b; 1 < x] \implies a < b$

$\langle\text{proof}\rangle$

**lemma** *powhr-less-cancel-iff* [*simp*]:

$1 < x \implies (x\ \text{powhr}\ a < x\ \text{powhr}\ b) = (a < b)$   
 $\langle\text{proof}\rangle$

**lemma** *powhr-le-cancel-iff* [*simp*]:

$1 < x \implies (x\ \text{powhr}\ a \leq x\ \text{powhr}\ b) = (a \leq b)$   
 $\langle\text{proof}\rangle$

**lemma** *hlog*:

$\text{hlog}\ (\text{star-}n\ X)\ (\text{star-}n\ Y) =$   
 $\text{star-}n\ (\%n.\ \text{log}\ (X\ n)\ (Y\ n))$   
 $\langle\text{proof}\rangle$

**lemma** *hlog-starfun-ln*:  $!!x.\ (*f* \ln)\ x = \text{hlog}\ ((*f* \exp)\ 1)\ x$

⟨proof⟩

**lemma** *powhr-hlog-cancel* [simp]:

!!a x. [| 0 < a; a ≠ 1; 0 < x |] ==> a powhr (hlog a x) = x  
 ⟨proof⟩

**lemma** *hlog-powhr-cancel* [simp]:

!!a y. [| 0 < a; a ≠ 1 |] ==> hlog a (a powhr y) = y  
 ⟨proof⟩

**lemma** *hlog-mult*:

!!a x y. [| 0 < a; a ≠ 1; 0 < x; 0 < y |]  
 ==> hlog a (x \* y) = hlog a x + hlog a y  
 ⟨proof⟩

**lemma** *hlog-as-starfun*:

!!a x. [| 0 < a; a ≠ 1 |] ==> hlog a x = (\*f\* ln) x / (\*f\* ln) a  
 ⟨proof⟩

**lemma** *hlog-eq-div-starfun-ln-mult-hlog*:

!!a b x. [| 0 < a; a ≠ 1; 0 < b; b ≠ 1; 0 < x |]  
 ==> hlog a x = ((\*f\* ln) b / (\*f\*ln) a) \* hlog b x  
 ⟨proof⟩

**lemma** *powhr-as-starfun*: !!a x. x powhr a = (\*f\* exp) (a \* (\*f\* ln) x)  
 ⟨proof⟩

**lemma** *HInfinite-powhr*:

[| x : HInfinite; 0 < x; a : HFinite – Infinitesimal;  
 0 < a |] ==> x powhr a : HInfinite  
 ⟨proof⟩

**lemma** *hlog-hrabs-HInfinite-Infinitesimal*:

[| x : HFinite – Infinitesimal; a : HInfinite; 0 < a |]  
 ==> hlog a (abs x) : Infinitesimal  
 ⟨proof⟩

**lemma** *hlog-HInfinite-as-starfun*:

[| a : HInfinite; 0 < a |] ==> hlog a x = (\*f\* ln) x / (\*f\* ln) a  
 ⟨proof⟩

**lemma** *hlog-one* [simp]: !!a. hlog a 1 = 0

⟨proof⟩

**lemma** *hlog-eq-one* [simp]: !!a. [| 0 < a; a ≠ 1 |] ==> hlog a a = 1

⟨proof⟩

**lemma** *hlog-inverse*:

[| 0 < a; a ≠ 1; 0 < x |] ==> hlog a (inverse x) = – hlog a x

*<proof>*

**lemma** *hlog-divide*:

$[[ 0 < a; a \neq 1; 0 < x; 0 < y ]] ==> \text{hlog } a (x/y) = \text{hlog } a x - \text{hlog } a y$   
*<proof>*

**lemma** *hlog-less-cancel-iff* [*simp*]:

$!!a x y. [[ 1 < a; 0 < x; 0 < y ]] ==> (\text{hlog } a x < \text{hlog } a y) = (x < y)$   
*<proof>*

**lemma** *hlog-le-cancel-iff* [*simp*]:

$[[ 1 < a; 0 < x; 0 < y ]] ==> (\text{hlog } a x \leq \text{hlog } a y) = (x \leq y)$   
*<proof>*

**end**

**theory** *Hyperreal*  
**imports** *Ln Deriv Taylor HLog*  
**begin**

**end**

**theory** *Hypercomplex*  
**imports** *CLim Hyperreal*  
**begin**

**end**