

# Countable Ordinals

Brian Huffman

December 17, 2016

## Abstract

This development defines a well-ordered type of countable ordinals. It includes notions of continuous and normal functions, recursively defined functions over ordinals, least fixed-points, and derivatives. Much of ordinal arithmetic is formalized, including exponentials and logarithms. The development concludes with formalizations of Cantor Normal Form and Veblen hierarchies over normal functions.

## Contents

<b>1</b>	<b>Definition of Ordinals</b>	<b>3</b>
1.1	Preliminary datatype for ordinals . . . . .	3
1.2	Ordinal type . . . . .	5
1.3	Induction over ordinals . . . . .	6
<b>2</b>	<b>Ordinal Induction</b>	<b>7</b>
2.1	Zero and successor ordinals . . . . .	8
2.1.1	Derived properties of 0 and oSuc . . . . .	8
2.2	Strict monotonicity . . . . .	9
2.3	Limit ordinals . . . . .	10
2.3.1	Making strict monotonic sequences . . . . .	12
2.4	Induction principle for ordinals . . . . .	12
<b>3</b>	<b>Continuity</b>	<b>13</b>
3.1	Continuous functions . . . . .	13
3.2	Normal functions . . . . .	13
<b>4</b>	<b>Recursive Definitions</b>	<b>14</b>
4.1	Partial orders . . . . .	15
4.2	Recursive definitions for <i>ordinal</i> $\Rightarrow$ <i>ordinal</i> . . . . .	16

<b>5</b>	<b>Ordinal Arithmetic</b>	<b>17</b>
5.1	Addition . . . . .	17
5.2	Subtraction . . . . .	18
5.3	Multiplication . . . . .	20
5.4	Exponentiation . . . . .	21
<b>6</b>	<b>Inverse Functions</b>	<b>23</b>
6.1	Division . . . . .	24
6.2	Derived properties of division . . . . .	26
6.3	Logarithms . . . . .	26
<b>7</b>	<b>Fixed-points</b>	<b>28</b>
7.1	Derivatives of ordinal functions . . . . .	29
<b>8</b>	<b>Omega</b>	<b>30</b>
8.1	Embedding naturals in the ordinals . . . . .	30
8.2	Omega, the least limit ordinal . . . . .	31
8.3	Arithmetic properties of $\omega$ . . . . .	31
8.4	Additive principal ordinals . . . . .	32
8.5	Cantor normal form . . . . .	33
8.6	Epsilon 0 . . . . .	35
<b>9</b>	<b>Veblen Hierarchies</b>	<b>35</b>
9.1	Closed, unbounded sets . . . . .	35
9.2	Ordering functions . . . . .	36
9.3	Critical ordinals . . . . .	37
9.4	Veblen hierarchy over a normal function . . . . .	37
9.5	Veblen hierarchy over $\lambda x. 1 + x$ . . . . .	38

# 1 Definition of Ordinals

**theory** *OrdinalDef*  
**imports** *Main*  
**begin**

## 1.1 Preliminary datatype for ordinals

**datatype** *ord0* = *ord0-Zero* | *ord0-Lim* *nat*  $\Rightarrow$  *ord0*

subterm ordering on *ord0*

**definition**

*ord0-prec* :: (*ord0*  $\times$  *ord0*) set **where**  
*ord0-prec* = ( $\bigcup$  *f i*. {(*f i*, *ord0-Lim f*)})

**lemma** *wf-ord0-prec*: *wf ord0-prec*

*<proof>*

**lemmas** *ord0-prec-induct* = *wf-induct*[*OF wf-trancl*[*OF wf-ord0-prec*]]

less-than-or-equal ordering on *ord0*

**inductive-set** *ord0-leq* :: (*ord0*  $\times$  *ord0*) set **where**

$\llbracket \forall a. (a, x) \in \text{ord0-prec}^+ \longrightarrow (\exists b. (b, y) \in \text{ord0-prec}^+ \wedge (a, b) \in \text{ord0-leq}) \rrbracket$   
 $\implies (x, y) \in \text{ord0-leq}$

**lemma** *ord0-leqI*:

$\llbracket \forall a. (a, x) \in \text{ord0-prec}^+ \longrightarrow (a, y) \in \text{ord0-leq} \ O \ \text{ord0-prec}^+ \rrbracket$   
 $\implies (x, y) \in \text{ord0-leq}$

*<proof>*

**lemma** *ord0-leqD*:

$\llbracket (x, y) \in \text{ord0-leq}; (a, x) \in \text{ord0-prec}^+ \rrbracket \implies (a, y) \in \text{ord0-leq} \ O \ \text{ord0-prec}^+$   
*<proof>*

**lemma** *ord0-leq-refl*:  $(x, x) \in \text{ord0-leq}$

*<proof>*

**lemma** *ord0-leq-trans*[*rule-format*]:

$\forall y. (x, y) \in \text{ord0-leq} \longrightarrow$   
 $(\forall z. (y, z) \in \text{ord0-leq} \longrightarrow (x, z) \in \text{ord0-leq})$

*<proof>*

**lemma** *wf-ord0-leq*: *wf (ord0-leq O ord0-prec<sup>+</sup>)*

*<proof>*

ordering on *ord0*

**instantiation** *ord0* :: *ord*

**begin**

**definition**

*ord0-less-def*:  $x < y \iff (x,y) \in \text{ord0-leq} \cup \text{ord0-prec}^+$

**definition**

*ord0-le-def*:  $x \leq y \iff (x,y) \in \text{ord0-leq}$

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

**end**

**lemma** *ord0-order-refl*[*simp*]:  $(x::\text{ord0}) \leq x$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-order-trans*:  $\llbracket (x::\text{ord0}) \leq y; y \leq z \rrbracket \implies x \leq z$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-wf*: *wf*  $\{(x,y::\text{ord0}). x < y\}$   
 $\langle \text{proof} \rangle$

**lemmas** *ord0-less-induct* = *wf-induct*[*OF ord0-wf*]

**lemma** *ord0-leI*:

$\llbracket \forall a::\text{ord0}. a < x \longrightarrow a < y \rrbracket \implies x \leq y$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-less-le-trans*:

$\llbracket (x::\text{ord0}) < y; y \leq z \rrbracket \implies x < z$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-le-less-trans*:

$\llbracket (x::\text{ord0}) \leq y; y < z \rrbracket \implies x < z$   
 $\langle \text{proof} \rangle$

**lemma** *rev-ord0-le-less-trans*:

$\llbracket (y::\text{ord0}) < z; x \leq y \rrbracket \implies x < z$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-less-trans*:

$\llbracket (x::\text{ord0}) < y; y < z \rrbracket \implies x < z$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-less-imp-le*:  $(x::\text{ord0}) < y \implies x \leq y$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-linear-lemma*:

**fixes**  $m :: \text{ord0}$  **and**  $n :: \text{ord0}$

**shows**  $m < n \vee n < m \vee (m \leq n \wedge n \leq m)$   
 $\langle \text{proof} \rangle$

**lemma** *ord0-linear*:  $(x::ord0) \leq y \vee y \leq x$   
*<proof>*

**lemma** *ord0-order-less-le*:  $(x::ord0) < y = (x \leq y \wedge \neg y \leq x)$   
*<proof>*

## 1.2 Ordinal type

### definition

*ord0rel* ::  $(ord0 \times ord0)$  set **where**  
*ord0rel* =  $\{(x,y). x \leq y \wedge y \leq x\}$

**typedef** *ordinal* =  $(UNIV::ord0$  set) // *ord0rel*  
*<proof>*

**theorem** *Abs-ordinal-cases2* [*case-names Abs-ordinal, cases type: ordinal*]:  
 $(\bigwedge z. x = \text{Abs-ordinal } (ord0rel \text{ `` } \{z\}) \implies P) \implies P$   
*<proof>*

**instantiation** *ordinal* :: ord  
**begin**

### definition

*ordinal-less-def*:  $x < y \longleftrightarrow (\forall a \in \text{Rep-ordinal } x. \forall b \in \text{Rep-ordinal } y. a < b)$

### definition

*ordinal-le-def*:  $x \leq y \longleftrightarrow (\forall a \in \text{Rep-ordinal } x. \forall b \in \text{Rep-ordinal } y. a \leq b)$

**instance** *<proof>*

**end**

**lemma** *Rep-Abs-ord0rel* [*simp*]:  
 $\text{Rep-ordinal } (\text{Abs-ordinal } (ord0rel \text{ `` } \{x\})) = (ord0rel \text{ `` } \{x\})$   
*<proof>*

**lemma** *mem-ord0rel-Image* [*simp, intro!*]:  $x \in ord0rel \text{ `` } \{x\}$   
*<proof>*

**lemma** *equiv-ord0rel*: *equiv UNIV ord0rel*  
*<proof>*

**lemma** *Abs-ordinal-eq* [*simp*]:  
 $(\text{Abs-ordinal } (ord0rel \text{ `` } \{x\}) = \text{Abs-ordinal } (ord0rel \text{ `` } \{y\}))$   
 $= (x \leq y \wedge y \leq x)$   
*<proof>*

**lemma** *Abs-ordinal-le* [*simp*]:

*Abs-ordinal* (*ord0rel* “ {*x*}) ≤ *Abs-ordinal* (*ord0rel* “ {*y*}) = (*x* ≤ *y*)  
⟨*proof*⟩

**lemma** *Abs-ordinal-less*[*simp*]:  
*Abs-ordinal* (*ord0rel* “ {*x*}) < *Abs-ordinal* (*ord0rel* “ {*y*}) = (*x* < *y*)  
⟨*proof*⟩

**lemma** *ordinal-order-refl*: (*x*::*ordinal*) ≤ *x*  
⟨*proof*⟩

**lemma** *ordinal-order-trans*: (*x*::*ordinal*) ≤ *y* ⇒ *y* ≤ *z* ⇒ *x* ≤ *z*  
⟨*proof*⟩

**lemma** *ordinal-order-antisym*: (*x*::*ordinal*) ≤ *y* ⇒ *y* ≤ *x* ⇒ *x* = *y*  
⟨*proof*⟩

**lemma** *ordinal-order-less-le-not-le*: ((*x*::*ordinal*) < *y*) = (*x* ≤ *y* ∧ ¬ *y* ≤ *x*)  
⟨*proof*⟩

**lemma** *ordinal-linear*: (*x*::*ordinal*) ≤ *y* ∨ *y* ≤ *x*  
⟨*proof*⟩

**lemma** *ordinal-wf*: *wf* {(*x,y*::*ordinal*). *x* < *y*}  
⟨*proof*⟩

**instance** *ordinal* :: *wellorder*  
⟨*proof*⟩

### 1.3 Induction over ordinals

zero and strict limits

**definition**  
*oZero* :: *ordinal* **where**  
*oZero* = *Abs-ordinal* (*ord0rel* “ {*ord0-Zero*})

**definition**  
*oStrictLimit* :: (*nat* ⇒ *ordinal*) ⇒ *ordinal* **where**  
*oStrictLimit* *f* = *Abs-ordinal*  
(*ord0rel* “ {*ord0-Lim* (λ*n*. *SOME* *x*. *x* ∈ *Rep-ordinal* (*f* *n*))})

induction over ordinals

**lemma** *ord0relD*: (*x,y*) ∈ *ord0rel* ⇒ *x* ≤ *y* ∧ *y* ≤ *x*  
⟨*proof*⟩

**lemma** *ord0-precD*: (*x,y*) ∈ *ord0-prec* ⇒ ∃ *f n*. *x* = *f n* ∧ *y* = *ord0-Lim* *f*  
⟨*proof*⟩

**lemma** *less-ord0-LimI*: *f n* < *ord0-Lim* *f*  
⟨*proof*⟩

**lemma** *less-ord0-LimD*:  $x < \text{ord0-Lim } f \implies \exists n. x \leq f\ n$   
⟨*proof*⟩

**lemma** *some-ord0rel*:  $(x, \text{SOME } y. (x,y) \in \text{ord0rel}) \in \text{ord0rel}$   
⟨*proof*⟩

**lemma** *ord0-Lim-le*:  
 $\forall n. f\ n \leq g\ n \implies \text{ord0-Lim } f \leq \text{ord0-Lim } g$   
⟨*proof*⟩

**lemma** *ord0-Lim-ord0rel*:  
 $\forall n. (f\ n, g\ n) \in \text{ord0rel} \implies (\text{ord0-Lim } f, \text{ord0-Lim } g) \in \text{ord0rel}$   
⟨*proof*⟩

**lemma** *Abs-ordinal-oStrictLimit*:  
*Abs-ordinal* (*ord0rel* “ {*ord0-Lim f*} )  
= *oStrictLimit* ( $\lambda n. \text{Abs-ordinal} (\text{ord0rel} \text{ “ } \{f\ n\})$ )  
⟨*proof*⟩

**lemma** *oStrictLimit-induct*:  
**assumes** *base*:  $P\ \text{oZero}$   
**assumes** *step*:  $\bigwedge f. \forall n. P\ (f\ n) \implies P\ (\text{oStrictLimit } f)$   
**shows**  $P\ a$   
⟨*proof*⟩

order properties of 0 and strict limits

**lemma** *oZero-least*:  $\text{oZero} \leq x$   
⟨*proof*⟩

**lemma** *oStrictLimit-ub*:  $f\ n < \text{oStrictLimit } f$   
⟨*proof*⟩

**lemma** *oStrictLimit-lub*:  $\forall n. f\ n < x \implies \text{oStrictLimit } f \leq x$   
⟨*proof*⟩

**lemma** *less-oStrictLimitD*:  $x < \text{oStrictLimit } f \implies \exists n. x \leq f\ n$   
⟨*proof*⟩

**end**

## 2 Ordinal Induction

**theory** *OrdinalInduct*  
**imports** *OrdinalDef*  
**begin**

## 2.1 Zero and successor ordinals

### definition

$oSuc :: ordinal \Rightarrow ordinal$  **where**  
 $oSuc\ x = oStrictLimit\ (\lambda n. x)$

**lemma** *less-oSuc* [iff]:  $x < oSuc\ x$   
<proof>

**lemma** *oSuc-leI*:  $x < y \implies oSuc\ x \leq y$   
<proof>

**instantiation** *ordinal* :: {zero, one}  
**begin**

### definition

*ordinal-zero-def*:  $(0::ordinal) = oZero$

### definition

*ordinal-one-def* [simp]:  $(1::ordinal) = oSuc\ 0$

**instance** <proof>

**end**

### 2.1.1 Derived properties of 0 and oSuc

**lemma** *less-oSuc-eq-le*:  $(x < oSuc\ y) = (x \leq y)$   
<proof>

**lemma** *ordinal-0-le* [iff]:  $0 \leq (x::ordinal)$   
<proof>

**lemma** *ordinal-not-less-0* [iff]:  $\neg (x::ordinal) < 0$   
<proof>

**lemma** *ordinal-le-0* [iff]:  $(x \leq 0) = (x = (0::ordinal))$   
<proof>

**lemma** *ordinal-neq-0* [iff]:  $(x \neq 0) = (0 < (x::ordinal))$   
<proof>

**lemma** *ordinal-not-0-less* [iff]:  $(\neg 0 < x) = (x = (0::ordinal))$   
<proof>

**lemma** *oSuc-le-eq-less*:  $(oSuc\ x \leq y) = (x < y)$   
<proof>

**lemma** *zero-less-oSuc* [iff]:  $0 < oSuc\ x$   
<proof>



**lemma** *oSuc-not-0* [iff]:  $oSuc\ x \neq 0$

*<proof>*

**lemma** *less-oSuc0* [iff]:  $(x < oSuc\ 0) = (x = 0)$

*<proof>*

**lemma** *oSuc-less-oSuc* [iff]:  $(oSuc\ x < oSuc\ y) = (x < y)$

*<proof>*

**lemma** *oSuc-eq-oSuc* [iff]:  $(oSuc\ x = oSuc\ y) = (x = y)$

*<proof>*

**lemma** *oSuc-le-oSuc* [iff]:  $(oSuc\ x \leq oSuc\ y) = (x \leq y)$

*<proof>*

**lemma** *le-oSucE*:

$\llbracket x \leq oSuc\ y; x < y \implies R; x = oSuc\ y \implies R \rrbracket \implies R$

*<proof>*

**lemma** *less-oSucE*:

$\llbracket x < oSuc\ y; x < y \implies P; x = y \implies P \rrbracket \implies P$

*<proof>*

## 2.2 Strict monotonicity

**locale** *strict-mono* =

**fixes** *f*

**assumes** *strict-mono*:  $A < B \implies f\ A < f\ B$

**lemmas** *strict-monoI* = *strict-mono.intro*

**and** *strict-monoD* = *strict-mono.strict-mono*

**lemma** *strict-mono-natI*:

**fixes**  $f :: nat \Rightarrow 'a::order$

**shows**  $(\bigwedge n. f\ n < f\ (Suc\ n)) \implies strict-mono\ f$

*<proof>*

**lemma** *mono-natI*:

**fixes**  $f :: nat \Rightarrow 'a::order$

**shows**  $(\bigwedge n. f\ n \leq f\ (Suc\ n)) \implies mono\ f$

*<proof>*

**lemma** *strict-mono-mono*:

**fixes**  $f :: 'a::order \Rightarrow 'b::order$

**shows**  $strict-mono\ f \implies mono\ f$

*<proof>*

**lemma** *strict-mono-monoD*:

**fixes**  $f :: 'a::order \Rightarrow 'b::order$   
**shows**  $\llbracket \text{strict-mono } f; A \leq B \rrbracket \Longrightarrow f A \leq f B$   
 $\langle \text{proof} \rangle$

**lemma** *strict-mono-cancel-eq*:  
**fixes**  $f :: 'a::linorder \Rightarrow 'b::linorder$   
**shows**  $\text{strict-mono } f \Longrightarrow (f x = f y) = (x = y)$   
 $\langle \text{proof} \rangle$

**lemma** *strict-mono-cancel-less*:  
**fixes**  $f :: 'a::linorder \Rightarrow 'b::linorder$   
**shows**  $\text{strict-mono } f \Longrightarrow (f x < f y) = (x < y)$   
 $\langle \text{proof} \rangle$

**lemma** *strict-mono-cancel-le*:  
**fixes**  $f :: 'a::linorder \Rightarrow 'b::linorder$   
**shows**  $\text{strict-mono } f \Longrightarrow (f x \leq f y) = (x \leq y)$   
 $\langle \text{proof} \rangle$

## 2.3 Limit ordinals

**definition**  
 $oLimit :: (nat \Rightarrow ordinal) \Rightarrow ordinal$  **where**  
 $oLimit f = (LEAST k. \forall n. f n \leq k)$

**lemma** *oLimit-leI*:  $\forall n. f n \leq x \Longrightarrow oLimit f \leq x$   
 $\langle \text{proof} \rangle$

**lemma** *le-oLimit [iff]*:  $f n \leq oLimit f$   
 $\langle \text{proof} \rangle$

**lemma** *le-oLimitI*:  $x \leq f n \Longrightarrow x \leq oLimit f$   
 $\langle \text{proof} \rangle$

**lemma** *less-oLimitI*:  $x < f n \Longrightarrow x < oLimit f$   
 $\langle \text{proof} \rangle$

**lemma** *less-oLimitD*:  $x < oLimit f \Longrightarrow \exists n. x < f n$   
 $\langle \text{proof} \rangle$

**lemma** *less-oLimitE*:  
 $\llbracket x < oLimit f; \bigwedge n. x < f n \Longrightarrow P \rrbracket \Longrightarrow P$   
 $\langle \text{proof} \rangle$

**lemma** *le-oLimitE*:  
 $\llbracket x \leq oLimit f; \bigwedge n. x \leq f n \Longrightarrow R; x = oLimit f \Longrightarrow R \rrbracket \Longrightarrow R$   
 $\langle \text{proof} \rangle$

**lemma** *oLimit-const [simp]*:  $oLimit (\lambda n. x) = x$

$\langle \text{proof} \rangle$

**lemma** *strict-mono-less-oLimit*:  
 $\text{strict-mono } f \implies f \ n < \text{oLimit } f$   
 $\langle \text{proof} \rangle$

**lemma** *oLimit-eqI*:  
 $\llbracket \bigwedge n. \exists m. f \ n \leq g \ m; \bigwedge n. \exists m. g \ n \leq f \ m \rrbracket \implies \text{oLimit } f = \text{oLimit } g$   
 $\langle \text{proof} \rangle$

**lemma** *oLimit-Suc*:  
 $f \ 0 < \text{oLimit } f \implies \text{oLimit } (\lambda n. f \ (\text{Suc } n)) = \text{oLimit } f$   
 $\langle \text{proof} \rangle$

**lemma** *oLimit-shift*:  
 $\forall n. f \ n < \text{oLimit } f \implies \text{oLimit } (\lambda n. f \ (n + k)) = \text{oLimit } f$   
 $\langle \text{proof} \rangle$

**lemma** *oLimit-shift-mono*:  
 $\text{mono } f \implies \text{oLimit } (\lambda n. f \ (n + k)) = \text{oLimit } f$   
 $\langle \text{proof} \rangle$

limit ordinal predicate

**definition**  
*limit-ordinal* :: *ordinal*  $\Rightarrow$  *bool* **where**  
*limit-ordinal*  $x \iff (x \neq 0) \wedge (\forall y. x \neq \text{oSuc } y)$

**lemma** *limit-ordinal-not-0* [simp]:  $\neg \text{limit-ordinal } 0$   
 $\langle \text{proof} \rangle$

**lemma** *zero-less-limit-ordinal* [simp]:  $\text{limit-ordinal } x \implies 0 < x$   
 $\langle \text{proof} \rangle$

**lemma** *limit-ordinal-not-oSuc* [simp]:  $\neg \text{limit-ordinal } (\text{oSuc } p)$   
 $\langle \text{proof} \rangle$

**lemma** *oSuc-less-limit-ordinal*:  
 $\text{limit-ordinal } x \implies (\text{oSuc } w < x) = (w < x)$   
 $\langle \text{proof} \rangle$

**lemma** *limit-ordinal-oLimitI*:  
 $\forall n. f \ n < \text{oLimit } f \implies \text{limit-ordinal } (\text{oLimit } f)$   
 $\langle \text{proof} \rangle$

**lemma** *strict-mono-limit-ordinal*:  
 $\text{strict-mono } f \implies \text{limit-ordinal } (\text{oLimit } f)$   
 $\langle \text{proof} \rangle$

**lemma** *limit-ordinalII*:

$\llbracket 0 < z; \forall x < z. \text{oSuc } x < z \rrbracket \implies \text{limit-ordinal } z$   
 ⟨proof⟩

### 2.3.1 Making strict monotonic sequences

**primrec** *make-mono* :: (nat  $\Rightarrow$  ordinal)  $\Rightarrow$  nat  $\Rightarrow$  nat  
**where**

*make-mono* f 0 = 0  
 | *make-mono* f (Suc n) = (LEAST x. f (make-mono f n) < f x)

**lemma** *f-make-mono-less*:

$\forall n. f n < \text{oLimit } f \implies f (\text{make-mono } f n) < f (\text{make-mono } f (\text{Suc } n))$   
 ⟨proof⟩

**lemma** *strict-mono-f-make-mono*:

$\forall n. f n < \text{oLimit } f \implies \text{strict-mono } (\lambda n. f (\text{make-mono } f n))$   
 ⟨proof⟩

**lemma** *le-f-make-mono*:

$\llbracket \forall n. f n < \text{oLimit } f; m \leq \text{make-mono } f n \rrbracket \implies f m \leq f (\text{make-mono } f n)$   
 ⟨proof⟩

**lemma** *make-mono-less*:

$\forall n. f n < \text{oLimit } f \implies \text{make-mono } f n < \text{make-mono } f (\text{Suc } n)$   
 ⟨proof⟩

**declare** *make-mono.simps* [simp del]

**lemma** *oLimit-make-mono-eq*:

$\forall n. f n < \text{oLimit } f \implies \text{oLimit } (\lambda n. f (\text{make-mono } f n)) = \text{oLimit } f$   
 ⟨proof⟩

## 2.4 Induction principle for ordinals

**lemma** *oLimit-le-oStrictLimit*:  $\text{oLimit } f \leq \text{oStrictLimit } f$   
 ⟨proof⟩

**lemma** *oLimit-induct*:

**assumes** *zero*:  $P 0$

**and** *suc*:  $\bigwedge x. P x \implies P (\text{oSuc } x)$

**and** *lim*:  $\bigwedge f. \llbracket \text{strict-mono } f; \forall n. P (f n) \rrbracket \implies P (\text{oLimit } f)$

**shows**  $P a$

⟨proof⟩

**lemma** *ordinal-cases*:

**assumes** *zero*:  $a = 0 \implies P$

**and** *suc*:  $\bigwedge x. a = \text{oSuc } x \implies P$

**and** *lim*:  $\bigwedge f. \llbracket \text{strict-mono } f; a = \text{oLimit } f \rrbracket \implies P$

**shows**  $P$

⟨proof⟩

end

### 3 Continuity

theory *OrdinalCont*  
imports *OrdinalInduct*  
begin

#### 3.1 Continuous functions

locale *continuous* =  
 fixes  $F :: \text{ordinal} \Rightarrow \text{ordinal}$   
 assumes *cont*:  $F (\text{oLimit } f) = \text{oLimit } (\lambda n. F (f n))$

lemmas *continuousD* = *continuous.cont*

lemma (in *continuous*) *mono*: *mono*  $F$   
*<proof>*

lemma (in *continuous*) *monoD*:  $x \leq y \Longrightarrow F x \leq F y$   
*<proof>*

lemma *continuousI*:  
 assumes *lim*:  $\bigwedge f. \text{strict-mono } f \Longrightarrow F (\text{oLimit } f) = \text{oLimit } (\lambda n. F (f n))$   
 assumes *suc*:  $\bigwedge x. F x \leq F (\text{oSuc } x)$   
 shows *continuous*  $F$   
*<proof>*

#### 3.2 Normal functions

locale *normal* = *continuous* +  
 assumes *strict*: *strict-mono*  $F$

lemma (in *normal*) *mono*: *mono*  $F$   
*<proof>*

lemma (in *normal*) *continuous*: *continuous*  $F$   
*<proof>*

lemma (in *normal*) *monoD*:  $x \leq y \Longrightarrow F x \leq F y$   
*<proof>*

lemma (in *normal*) *strict-monoD*:  $x < y \Longrightarrow F x < F y$   
*<proof>*

lemma (in *normal*) *cancel-eq*:  $(F x = F y) = (x = y)$   
*<proof>*

**lemma** (in normal) *cancel-less*:  $(F x < F y) = (x < y)$   
⟨proof⟩

**lemma** (in normal) *cancel-le*:  $(F x \leq F y) = (x \leq y)$   
⟨proof⟩

**lemma** (in normal) *oLimit*:  $F (oLimit f) = oLimit (\lambda n. F (f n))$   
⟨proof⟩

**lemma** (in normal) *increasing*:  $x \leq F x$   
⟨proof⟩

**lemma** *normalI*:  
**assumes** *lim*:  $\bigwedge f. \text{strict-mono } f \implies F (oLimit f) = oLimit (\lambda n. F (f n))$   
**assumes** *suc*:  $\bigwedge x. F x < F (oSuc x)$   
**shows** *normal F*  
⟨proof⟩

**lemma** *normal-range-le*:  
[[normal F; normal G; range G  $\subseteq$  range F]]  $\implies F x \leq G x$   
⟨proof⟩

**lemma** *normal-range-eq*:  
[[normal F; normal G; range F = range G]]  $\implies F = G$   
⟨proof⟩

end

## 4 Recursive Definitions

**theory** *OrdinalRec*  
**imports** *OrdinalCont*  
**begin**

**definition**  
*oPrec* :: *ordinal*  $\Rightarrow$  *ordinal* **where**  
*oPrec*  $x = (THE p. x = oSuc p)$

**lemma** *oPrec-oSuc [simp]*:  $oPrec (oSuc x) = x$   
⟨proof⟩

**lemma** *oPrec-less*:  $\exists p. x = oSuc p \implies oPrec x < x$   
⟨proof⟩

**definition**  
*ordinal-rec0* ::  
[*'a*, *ordinal*  $\Rightarrow$  *'a*  $\Rightarrow$  *'a*, (*nat*  $\Rightarrow$  *'a*)  $\Rightarrow$  *'a*, *ordinal*]  $\Rightarrow$  *'a* **where**  
*ordinal-rec0*  $z s l \equiv wfrec \{(x,y). x < y\} (\lambda F x.$

*if*  $x = 0$  *then*  $z$  *else*  
*if*  $(\exists p. x = \text{oSuc } p)$  *then*  $s (\text{oPrec } x) (F (\text{oPrec } x))$  *else*  
 $(\text{THE } y. \forall f. (\forall n. f \ n < \text{oLimit } f) \wedge \text{oLimit } f = x$   
 $\longrightarrow l (\lambda n. F (f \ n)) = y)$

**lemma** *ordinal-rec0-0:*

*ordinal-rec0*  $z \ s \ l \ 0 = z$

*<proof>*

**lemma** *ordinal-rec0-oSuc:*

*ordinal-rec0*  $z \ s \ l (\text{oSuc } x) = s \ x (\text{ordinal-rec0 } z \ s \ l \ x)$

*<proof>*

**lemma** *limit-ordinal-not-0:* *limit-ordinal*  $x \implies x \neq 0$

*<proof>*

**lemma** *limit-ordinal-not-oSuc:* *limit-ordinal*  $x \implies x \neq \text{oSuc } p$

*<proof>*

**lemma** *ordinal-rec0-limit-ordinal:*

*limit-ordinal*  $x \implies \text{ordinal-rec0 } z \ s \ l \ x =$

$(\text{THE } y. \forall f. (\forall n. f \ n < \text{oLimit } f) \wedge \text{oLimit } f = x \longrightarrow$

$l (\lambda n. \text{ordinal-rec0 } z \ s \ l (f \ n)) = y)$

*<proof>*

## 4.1 Partial orders

**locale** *porder* =

**fixes**  $le :: 'a \Rightarrow 'a \Rightarrow \text{bool}$  (**infixl**  $<<$  55)

**assumes** *po-refl*:  $\bigwedge x. x << x$

**and** *po-trans*:  $\bigwedge x \ y \ z. \llbracket x << y; y << z \rrbracket \implies x << z$

**and** *po-antisym*:  $\bigwedge x \ y. \llbracket x << y; y << x \rrbracket \implies x = y$

**lemma** *porder-order*: *porder*  $(\text{op } \le :: 'a :: \text{order} \Rightarrow 'a \Rightarrow \text{bool})$

*<proof>*

**lemma** (**in** *porder*) *flip*: *porder*  $(\lambda x \ y. y << x)$

*<proof>*

**locale** *omega-complete* = *porder* +

**fixes**  $\text{lub} :: (\text{nat} \Rightarrow 'a) \Rightarrow 'a$

**assumes** *is-ub-lub*:  $\bigwedge f \ n. f \ n << \text{lub } f$

**assumes** *is-lub-lub*:  $\bigwedge f \ x. \forall n. f \ n << x \implies \text{lub } f << x$

**lemma** (**in** *omega-complete*) *lub-cong-lemma*:

$\llbracket \forall n. f \ n < \text{oLimit } f; \forall m. g \ m < \text{oLimit } g; \text{oLimit } f \leq \text{oLimit } g;$

$\forall y < \text{oLimit } g. \forall x \leq y. F \ x << F \ y \rrbracket$

$\implies \text{lub } (\lambda n. F (f \ n)) << \text{lub } (\lambda n. F (g \ n))$

*<proof>*

**lemma** (in *omega-complete*) *lub-cong*:  
 $\llbracket \forall n. f\ n < oLimit\ f; \forall m. g\ m < oLimit\ g; oLimit\ f = oLimit\ g; \forall y < oLimit\ g. \forall x \leq y. F\ x << F\ y \rrbracket$   
 $\implies lub\ (\lambda n. F\ (f\ n)) = lub\ (\lambda n. F\ (g\ n))$   
 ⟨proof⟩

**lemma** (in *omega-complete*) *ordinal-rec0-mono-lemma*:  
**assumes**  $s: \forall p\ x. x << s\ p\ x$   
**shows**  $\forall y \leq w. \forall x \leq y. ordinal-rec0\ z\ s\ lub\ x << ordinal-rec0\ z\ s\ lub\ y$   
 ⟨proof⟩

**lemma** (in *omega-complete*) *ordinal-rec0-mono*:  
**assumes**  $s: \forall p\ x. x << s\ p\ x$   
**shows**  $x \leq y \implies ordinal-rec0\ z\ s\ lub\ x << ordinal-rec0\ z\ s\ lub\ y$   
 ⟨proof⟩

**lemma** (in *omega-complete*) *ordinal-rec0-oLimit*:  
**assumes**  $s: \forall p\ x. x << s\ p\ x$   
**shows**  $ordinal-rec0\ z\ s\ lub\ (oLimit\ f) =$   
 $lub\ (\lambda n. ordinal-rec0\ z\ s\ lub\ (f\ n))$   
 ⟨proof⟩

## 4.2 Recursive definitions for *ordinal* $\Rightarrow$ *ordinal*

**definition**  
*ordinal-rec* ::  
 $[ordinal, ordinal \Rightarrow ordinal \Rightarrow ordinal, ordinal] \Rightarrow ordinal$  **where**  
*ordinal-rec*  $z\ s = ordinal-rec0\ z\ s\ oLimit$

**lemma** *omega-complete-oLimit*: *omega-complete* ( $op \leq$ ) *oLimit*  
 ⟨proof⟩

**lemma** *ordinal-rec-0 [simp]*: *ordinal-rec*  $z\ s\ 0 = z$   
 ⟨proof⟩

**lemma** *ordinal-rec-oSuc [simp]*:  
*ordinal-rec*  $z\ s\ (oSuc\ x) = s\ x\ (ordinal-rec\ z\ s\ x)$   
 ⟨proof⟩

**lemma** *ordinal-rec-oLimit*:  
**assumes**  $s: \forall p\ x. x \leq s\ p\ x$   
**shows**  $ordinal-rec\ z\ s\ (oLimit\ f) = oLimit\ (\lambda n. ordinal-rec\ z\ s\ (f\ n))$   
 ⟨proof⟩

**lemma** *continuous-ordinal-rec*:  
**assumes**  $s: \forall p\ x. x \leq s\ p\ x$   
**shows** *continuous* (*ordinal-rec*  $z\ s$ )  
 ⟨proof⟩



**lemma** *mono-ordinal-rec*:  
**assumes**  $s: \forall p x. x \leq s p x$   
**shows** *mono* (*ordinal-rec*  $z s$ )  
*<proof>*

**lemma** *normal-ordinal-rec*:  
**assumes**  $s: \forall p x. x < s p x$   
**shows** *normal* (*ordinal-rec*  $z s$ )  
*<proof>*

**end**

## 5 Ordinal Arithmetic

**theory** *OrdinalArith*  
**imports** *OrdinalRec*  
**begin**

### 5.1 Addition

**instantiation** *ordinal* :: *plus*  
**begin**

**definition**  
 $op + = (\lambda x. \text{ordinal-rec } x (\lambda p. oSuc))$

**instance** *<proof>*

**end**

**lemma** *normal-plus: normal* ( $op + x$ )  
*<proof>*

**lemma** *ordinal-plus-0* [*simp*]:  $x + 0 = (x::\text{ordinal})$   
*<proof>*

**lemma** *ordinal-plus-oSuc* [*simp*]:  $x + oSuc y = oSuc (x + y)$   
*<proof>*

**lemma** *ordinal-plus-oLimit* [*simp*]:  $x + oLimit f = oLimit (\lambda n. x + f n)$   
*<proof>*

**lemma** *ordinal-0-plus* [*simp*]:  $0 + x = (x::\text{ordinal})$   
*<proof>*

**lemma** *ordinal-plus-assoc*:  
 $(x + y) + z = x + (y + z::\text{ordinal})$   
*<proof>*

**lemma** *ordinal-plus-monoL* [rule-format]:

$\forall x x'. x \leq x' \longrightarrow x + y \leq x' + (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-monoR*:  $y \leq y' \Longrightarrow x + y \leq x + (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-mono*:

$\llbracket x \leq x'; y \leq y' \rrbracket \Longrightarrow x + y \leq x' + (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-strict-monoR*:  $y < y' \Longrightarrow x + y < x + (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-le-plusL* [simp]:  $y \leq x + (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-le-plusR* [simp]:  $x \leq x + (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-less-plusR*:  $0 < y \Longrightarrow x < x + (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-left-cancel* [simp]:  
 $(w + x = w + y) = (x = (y::\text{ordinal}))$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-left-cancel-le* [simp]:  
 $(w + x \leq w + y) = (x \leq (y::\text{ordinal}))$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-left-cancel-less* [simp]:  
 $(w + x < w + y) = (x < (y::\text{ordinal}))$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-not-0*:  $(0 < x + y) = (0 < x \vee 0 < (y::\text{ordinal}))$   
 $\langle \text{proof} \rangle$

**lemma** *not-inject*:  $(\neg P) = (\neg Q) \Longrightarrow P = Q$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-plus-eq-0*:  
 $((x::\text{ordinal}) + y = 0) = (x = 0 \wedge y = 0)$   
 $\langle \text{proof} \rangle$

## 5.2 Subtraction

**instantiation** *ordinal* :: *minus*

**begin**

**definition**

*minus-ordinal-def*:

$x - y = \text{ordinal-rec } 0 \ (\lambda p \ w. \ \text{if } y \leq p \ \text{then } \text{oSuc } w \ \text{else } w) \ x$

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

**end**

**lemma** *continuous-minus*: *continuous*  $(\lambda x. \ x - y)$

$\langle \text{proof} \rangle$

**lemma** *ordinal-0-minus* [simp]:  $0 - x = (0::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-oSuc-minus* [simp]:  $y \leq x \implies \text{oSuc } x - y = \text{oSuc } (x - y)$

$\langle \text{proof} \rangle$

**lemma** *ordinal-oLimit-minus* [simp]:  $\text{oLimit } f - y = \text{oLimit } (\lambda n. \ f \ n - y)$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minus-0* [simp]:  $x - 0 = (x::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-oSuc-minus2*:  $x < y \implies \text{oSuc } x - y = x - y$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minus-eq-0* [rule-format, simp]:

$x \leq y \longrightarrow x - y = (0::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-plus-minus1* [simp]:  $(x + y) - x = (y::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-plus-minus2* [simp]:  $x \leq y \implies x + (y - x) = (y::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minusI*:  $x = y + z \implies x - y = (z::\text{ordinal})$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minus-less-eq* [simp]:

$(y::\text{ordinal}) \leq x \implies (x - y < z) = (x < y + z)$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minus-le-eq* [simp]:

$(x - y \leq z) = (x \leq y + (z::\text{ordinal}))$

$\langle \text{proof} \rangle$

**lemma** *ordinal-minus-monoL*:  $x \leq y \implies x - z \leq y - (z::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-minus-monoR*:  $x \leq y \implies z - y \leq z - (x::\text{ordinal})$   
(*proof*)

### 5.3 Multiplication

**instantiation** *ordinal* :: *times*  
**begin**

**definition**

*times-ordinal-def*:  $op * = (\lambda x. \text{ordinal-rec } 0 (\lambda p w. w + x))$

**instance** (*proof*)

**end**

**lemma** *continuous-times*: *continuous* ( $op * x$ )  
(*proof*)

**lemma** *normal-times*:  $0 < x \implies \text{normal } (op * x)$   
(*proof*)

**lemma** *ordinal-times-0* [*simp*]:  $x * 0 = (0::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-times-oSuc* [*simp*]:  $x * oSuc y = (x * y) + x$   
(*proof*)

**lemma** *ordinal-times-oLimit* [*simp*]:  $x * oLimit f = oLimit (\lambda n. x * f n)$   
(*proof*)

**lemma** *ordinal-0-times* [*simp*]:  $0 * x = (0::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-1-times* [*simp*]:  $oSuc 0 * x = (x::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-times-1* [*simp*]:  $x * oSuc 0 = (x::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-times-distrib*:

$x * (y + z) = (x * y) + (x * z::\text{ordinal})$   
(*proof*)

**lemma** *ordinal-times-assoc*:

$(x * y::\text{ordinal}) * z = x * (y * z)$   
(*proof*)

**lemma** *ordinal-times-monoL* [rule-format]:

$\forall x x'. x \leq x' \longrightarrow x * y \leq x' * (y::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-times-monoR*:  $y \leq y' \Longrightarrow x * y \leq x * (y'::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-times-mono*:

$\llbracket x \leq x'; y \leq y' \rrbracket \Longrightarrow x * y \leq x' * (y'::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-times-strict-monoR*:

$\llbracket y < y'; 0 < x \rrbracket \Longrightarrow x * y < x * (y'::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-le-timesL* [simp]:  $0 < x \Longrightarrow y \leq x * (y::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-le-timesR* [simp]:  $0 < y \Longrightarrow x \leq x * (y::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-less-timesR*:  $\llbracket 0 < x; oSuc\ 0 < y \rrbracket \Longrightarrow x < x * (y::ordinal)$   
 $\langle proof \rangle$

**lemma** *ordinal-times-left-cancel* [simp]:

$0 < w \Longrightarrow (w * x = w * y) = (x = (y::ordinal))$   
 $\langle proof \rangle$

**lemma** *ordinal-times-left-cancel-le* [simp]:

$0 < w \Longrightarrow (w * x \leq w * y) = (x \leq (y::ordinal))$   
 $\langle proof \rangle$

**lemma** *ordinal-times-left-cancel-less* [simp]:

$0 < w \Longrightarrow (w * x < w * y) = (x < (y::ordinal))$   
 $\langle proof \rangle$

**lemma** *ordinal-times-eq-0*:

$((x::ordinal) * y = 0) = (x = 0 \vee y = 0)$   
 $\langle proof \rangle$

**lemma** *ordinal-times-not-0* [simp]:

$((0::ordinal) < x * y) = (0 < x \wedge 0 < y)$   
 $\langle proof \rangle$

## 5.4 Exponentiation

**definition**

*exp-ordinal* :: [ordinal, ordinal]  $\Rightarrow$  ordinal (**infixr** \*\* 75) **where**

$op ** = (\lambda x. \text{if } 0 < x \text{ then ordinal-rec } 1 (\lambda p w. w * x) \\ \text{else } (\lambda y. \text{if } y = 0 \text{ then } 1 \text{ else } 0))$

**lemma** *continuous-exp*:  $0 < x \implies \text{continuous } (op ** x)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-0* [*simp*]:  $x ** 0 = (1::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-oSuc* [*simp*]:  $x ** oSuc y = (x ** y) * x$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-oLimit* [*simp*]:  
 $0 < x \implies x ** oLimit f = oLimit (\lambda n. x ** f n)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-0-exp* [*simp*]:  $0 ** x = (\text{if } x = 0 \text{ then } 1 \text{ else } 0)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-1-exp* [*simp*]:  $oSuc 0 ** x = oSuc 0$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-1* [*simp*]:  $x ** oSuc 0 = x$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-distrib*:  
 $x ** (y + z) = (x ** y) * (x ** (z::\text{ordinal}))$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-not-0* [*simp*]:  $(0 < x ** y) = (0 < x \vee y = 0)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-eq-0* [*simp*]:  $(x ** y = 0) = (x = 0 \wedge 0 < y)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-assoc*:  
 $(x ** y) ** z = x ** (y * z)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-monoL* [*rule-format*]:  
 $\forall x x'. x \leq x' \longrightarrow x ** y \leq x' ** (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *normal-exp*:  $oSuc 0 < x \implies \text{normal } (op ** x)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-monoR*:  
 $\llbracket 0 < x; y \leq y' \rrbracket \implies x ** y \leq x ** (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-mono*:  
 $\llbracket 0 < x'; x \leq x'; y \leq y' \rrbracket \implies x ** y \leq x' ** (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-strict-monoR*:  
 $\llbracket \text{oSuc } 0 < x; y < y' \rrbracket \implies x ** y < x ** (y'::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-le-expR* [simp]:  $0 < y \implies x \leq x ** (y::\text{ordinal})$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-left-cancel* [simp]:  
 $\text{oSuc } 0 < w \implies (w ** x = w ** y) = (x = y)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-left-cancel-le* [simp]:  
 $\text{oSuc } 0 < w \implies (w ** x \leq w ** y) = (x \leq y)$   
 $\langle \text{proof} \rangle$

**lemma** *ordinal-exp-left-cancel-less* [simp]:  
 $\text{oSuc } 0 < w \implies (w ** x < w ** y) = (x < y)$   
 $\langle \text{proof} \rangle$

**end**

## 6 Inverse Functions

**theory** *OrdinalInverse*  
**imports** *OrdinalArith*  
**begin**

**lemma** (**in normal**) *oInv-ex*:  
 $F 0 \leq a \implies \exists q. F q \leq a \wedge a < F (\text{oSuc } q)$   
 $\langle \text{proof} \rangle$

**lemma** *oInv-uniq*:  
 $\llbracket \text{mono } (F::\text{ordinal} \Rightarrow \text{ordinal});$   
 $F x \leq a \wedge a < F (\text{oSuc } x); F y \leq a \wedge a < F (\text{oSuc } y) \rrbracket$   
 $\implies x = y$   
 $\langle \text{proof} \rangle$

**definition**  
 $\text{oInv} :: (\text{ordinal} \Rightarrow \text{ordinal}) \Rightarrow \text{ordinal} \Rightarrow \text{ordinal}$  **where**  
 $\text{oInv } F a = (\text{if } F 0 \leq a \text{ then } (\text{THE } x. F x \leq a \wedge a < F (\text{oSuc } x)) \text{ else } 0)$

**lemma** (**in normal**) *oInv-bounds*:  
 $F 0 \leq a \implies F (\text{oInv } F a) \leq a \wedge a < F (\text{oSuc } (\text{oInv } F a))$   
 $\langle \text{proof} \rangle$

**lemma** (in normal) *oInv-bound1*:

$F\ 0 \leq a \implies F\ (oInv\ F\ a) \leq a$

*<proof>*

**lemma** (in normal) *oInv-bound2*:

$a < F\ (oSuc\ (oInv\ F\ a))$

*<proof>*

**lemma** (in normal) *oInv-equality*:

$\llbracket F\ x \leq a; a < F\ (oSuc\ x) \rrbracket \implies oInv\ F\ a = x$

*<proof>*

**lemma** (in normal) *oInv-inverse*:  $oInv\ F\ (F\ x) = x$

*<proof>*

**lemma** (in normal) *oInv-equality'*:  $a = F\ x \implies oInv\ F\ a = x$

*<proof>*

**lemma** (in normal) *oInv-eq-0*:  $a \leq F\ 0 \implies oInv\ F\ a = 0$

*<proof>*

**lemma** (in normal) *oInv-less*:

$\llbracket F\ 0 \leq a; a < F\ z \rrbracket \implies oInv\ F\ a < z$

*<proof>*

**lemma** (in normal) *le-oInv*:

$F\ z \leq a \implies z \leq oInv\ F\ a$

*<proof>*

**lemma** (in normal) *less-oInvD*:

$x < oInv\ F\ a \implies F\ (oSuc\ x) \leq a$

*<proof>*

**lemma** (in normal) *oInv-le*:

$a < F\ (oSuc\ x) \implies oInv\ F\ a \leq x$

*<proof>*

**lemma** (in normal) *mono-oInv*: *mono* (*oInv* *F*)

*<proof>*

**lemma** (in normal) *oInv-decreasing*:

$F\ 0 \leq x \implies oInv\ F\ x \leq x$

*<proof>*

## 6.1 Division

**instantiation** *ordinal* :: *modulo*

**begin**



**definition***div-ordinal-def:* $x \text{ div } y = (\text{if } 0 < y \text{ then } \text{oInv } (op * y) x \text{ else } 0)$ **definition***mod-ordinal-def:* $x \text{ mod } y = ((x::\text{ordinal}) - y * (x \text{ div } y))$ **instance**  $\langle \text{proof} \rangle$ **end****lemma** *ordinal-divI*:  $\llbracket x = y * q + r; r < y \rrbracket \implies x \text{ div } y = (q::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-times-div-le*:  $y * (x \text{ div } y) \leq (x::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-less-times-div-plus*: $0 < y \implies x < y * (x \text{ div } y) + (y::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-modI*:  $\llbracket x = y * q + r; r < y \rrbracket \implies x \text{ mod } y = (r::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-mod-less*:  $0 < y \implies x \text{ mod } y < (y::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-div-plus-mod*:  $y * (x \text{ div } y) + (x \text{ mod } y) = (x::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-div-less*:  $x < y * z \implies x \text{ div } y < (z::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-le-div*:  $\llbracket 0 < y; y * z \leq x \rrbracket \implies (z::\text{ordinal}) \leq x \text{ div } y$  $\langle \text{proof} \rangle$ **lemma** *ordinal-mono-div*:  $\text{mono } (\lambda x. x \text{ div } y::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-div-monoL*:  $x \leq x' \implies x \text{ div } y \leq x' \text{ div } (y::\text{ordinal})$  $\langle \text{proof} \rangle$ **lemma** *ordinal-div-decreasing*:  $(x::\text{ordinal}) \text{ div } y \leq x$  $\langle \text{proof} \rangle$ **lemma** *ordinal-div-0*:  $x \text{ div } 0 = (0::\text{ordinal})$  $\langle \text{proof} \rangle$

**lemma** *ordinal-mod-0*:  $x \text{ mod } 0 = (x::\text{ordinal})$   
(proof)

## 6.2 Derived properties of division

**lemma** *ordinal-div-1* [simp]:  $x \text{ div } \text{oSuc } 0 = x$   
(proof)

**lemma** *ordinal-mod-1* [simp]:  $x \text{ mod } \text{oSuc } 0 = 0$   
(proof)

**lemma** *ordinal-div-self* [simp]:  $0 < x \implies x \text{ div } x = (1::\text{ordinal})$   
(proof)

**lemma** *ordinal-mod-self* [simp]:  $x \text{ mod } x = (0::\text{ordinal})$   
(proof)

**lemma** *ordinal-div-greater* [simp]:  $x < y \implies x \text{ div } y = (0::\text{ordinal})$   
(proof)

**lemma** *ordinal-mod-greater* [simp]:  $x < y \implies x \text{ mod } y = (x::\text{ordinal})$   
(proof)

**lemma** *ordinal-0-div* [simp]:  $0 \text{ div } x = (0::\text{ordinal})$   
(proof)

**lemma** *ordinal-0-mod* [simp]:  $0 \text{ mod } x = (0::\text{ordinal})$   
(proof)

**lemma** *ordinal-1-dvd* [simp]:  $\text{oSuc } 0 \text{ dvd } x$   
(proof)

**lemma** *ordinal-dvd-mod*:  $y \text{ dvd } x = (x \text{ mod } y = (0::\text{ordinal}))$   
(proof)

**lemma** *ordinal-dvd-times-div*:  
 $y \text{ dvd } x \implies y * (x \text{ div } y) = (x::\text{ordinal})$   
(proof)

**lemma** *ordinal-dvd-oLimit*:  $\forall n. x \text{ dvd } f n \implies x \text{ dvd } \text{oLimit } f$   
(proof)

## 6.3 Logarithms

**definition**

$\text{oLog} :: \text{ordinal} \Rightarrow \text{ordinal} \Rightarrow \text{ordinal}$  **where**  
 $\text{oLog } b = (\lambda x. \text{if } 1 < b \text{ then } \text{oInv } (\text{op } ** b) x \text{ else } 0)$

**lemma** *ordinal-oLogI*:

$\llbracket b ** y \leq x; x < b ** y * b \rrbracket \implies oLog\ b\ x = y$   
 $\langle proof \rangle$

**lemma** *ordinal-exp-oLog-le*:  
 $\llbracket 0 < x; oSuc\ 0 < b \rrbracket \implies b ** (oLog\ b\ x) \leq x$   
 $\langle proof \rangle$

**lemma** *ordinal-less-exp-oLog*:  
 $oSuc\ 0 < b \implies x < b ** (oLog\ b\ x) * b$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-less*:  
 $\llbracket 0 < x; oSuc\ 0 < b; x < b ** y \rrbracket \implies oLog\ b\ x < y$   
 $\langle proof \rangle$

**lemma** *ordinal-le-oLog*:  
 $\llbracket oSuc\ 0 < b; b ** y \leq x \rrbracket \implies y \leq oLog\ b\ x$   
 $\langle proof \rangle$

**lemma** *ordinal-oLogI2*:  
 $\llbracket oSuc\ 0 < b; x = b ** y * q + r; 0 < q; q < b; r < b ** y \rrbracket \implies oLog\ b\ x = y$   
 $\langle proof \rangle$

**lemma** *ordinal-div-exp-oLog-less*:  
 $oSuc\ 0 < b \implies x\ div\ (b ** oLog\ b\ x) < b$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-base-0*:  $oLog\ 0\ x = 0$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-base-1*:  $oLog\ (oSuc\ 0)\ x = 0$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-0*:  $oLog\ b\ 0 = 0$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-exp*:  $oSuc\ 0 < b \implies oLog\ b\ (b ** x) = x$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-self*:  $oSuc\ 0 < b \implies oLog\ b\ b = oSuc\ 0$   
 $\langle proof \rangle$

**lemma** *ordinal-mono-oLog*:  $mono\ (oLog\ b)$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-monoR*:  $x \leq y \implies oLog\ b\ x \leq oLog\ b\ y$   
 $\langle proof \rangle$

**lemma** *ordinal-oLog-decreasing*:  $oLog\ b\ x \leq x$

⟨proof⟩

end

## 7 Fixed-points

**theory** *OrdinalFix*  
**imports** *OrdinalInverse*  
**begin**

**primrec** *iter* :: *nat*  $\Rightarrow$  (*'a*  $\Rightarrow$  *'a*)  $\Rightarrow$  (*'a*  $\Rightarrow$  *'a*)

**where**

*iter* 0  $F$  *x* = *x*  
| *iter* (*Suc* *n*) *F* *x* = *F* (*iter* *n* *F* *x*)

**definition**

*oFix* :: (*ordinal*  $\Rightarrow$  *ordinal*)  $\Rightarrow$  *ordinal*  $\Rightarrow$  *ordinal* **where**  
*oFix* *F* *a* = *oLimit* ( $\lambda$ *n.* *iter* *n* *F* *a*)

**lemma** *oFix-fixed*:

$\llbracket \text{continuous } F; a \leq F a \rrbracket \Longrightarrow F (oFix F a) = oFix F a$   
⟨proof⟩

**lemma** *oFix-least*:

$\llbracket \text{mono } F; F x = x; a \leq x \rrbracket \Longrightarrow oFix F a \leq x$   
⟨proof⟩

**lemma** *mono-oFix*: *mono* *F*  $\Longrightarrow$  *mono* (*oFix* *F*)

⟨proof⟩

**lemma** *less-oFixD*:

$\llbracket x < oFix F a; \text{mono } F; F x = x \rrbracket \Longrightarrow x < a$   
⟨proof⟩

**lemma** *less-oFixI*: *a* < *F* *a*  $\Longrightarrow$  *a* < *oFix* *F* *a*

⟨proof⟩

**lemma** *le-oFix*: *a*  $\leq$  *oFix* *F* *a*

⟨proof⟩

**lemma** *le-oFix1*: *F* *a*  $\leq$  *oFix* *F* *a*

⟨proof⟩

**lemma** *less-oFix-0D*:

$\llbracket x < oFix F 0; \text{mono } F \rrbracket \Longrightarrow x < F x$   
⟨proof⟩

**lemma** *zero-less-oFix-eq*: (*0* < *oFix* *F* *0*) = (*0* < *F* *0*)

⟨proof⟩

**lemma** *oFix-eq-self*:  $F a = a \implies oFix F a = a$   
 ⟨proof⟩

## 7.1 Derivatives of ordinal functions

The derivative of  $F$  enumerates all the fixed-points of  $F$

**definition**

$oDeriv :: (ordinal \Rightarrow ordinal) \Rightarrow ordinal \Rightarrow ordinal$  **where**  
 $oDeriv F = ordinal-rec (oFix F 0) (\lambda p x. oFix F (oSuc x))$

**lemma** *oDeriv-0 [simp]*:  
 $oDeriv F 0 = oFix F 0$   
 ⟨proof⟩

**lemma** *oDeriv-oSuc [simp]*:  
 $oDeriv F (oSuc x) = oFix F (oSuc (oDeriv F x))$   
 ⟨proof⟩

**lemma** *oDeriv-oLimit [simp]*:  
 $oDeriv F (oLimit f) = oLimit (\lambda n. oDeriv F (f n))$   
 ⟨proof⟩

**lemma** *oDeriv-fixed*:  
 $normal F \implies F (oDeriv F n) = oDeriv F n$   
 ⟨proof⟩

**lemma** *oDeriv-fixedD*:  
 $\llbracket oDeriv F x = x; normal F \rrbracket \implies F x = x$   
 ⟨proof⟩

**lemma** *normal-oDeriv*:  
 $normal (oDeriv F)$   
 ⟨proof⟩

**lemma** *oDeriv-increasing*:  
 $continuous F \implies F x \leq oDeriv F x$   
 ⟨proof⟩

**lemma** *oDeriv-total*:  
 $\llbracket normal F; F x = x \rrbracket \implies \exists n. x = oDeriv F n$   
 ⟨proof⟩

**lemma** *range-oDeriv*:  
 $normal F \implies range (oDeriv F) = \{x. F x = x\}$   
 ⟨proof⟩

**end**

## 8 Omega

```
theory OrdinalOmega
imports OrdinalFix
begin
```

### 8.1 Embedding naturals in the ordinals

```
primrec ordinal-of-nat :: nat  $\Rightarrow$  ordinal
where
```

```
  ordinal-of-nat 0 = 0
| ordinal-of-nat (Suc n) = oSuc (ordinal-of-nat n)
```

```
lemma strict-mono-ordinal-of-nat: strict-mono ordinal-of-nat
<proof>
```

```
lemma not-limit-ordinal-nat:  $\neg$  limit-ordinal (ordinal-of-nat n)
<proof>
```

```
lemma ordinal-of-nat-eq [simp]:
(ordinal-of-nat x = ordinal-of-nat y) = (x = y)
<proof>
```

```
lemma ordinal-of-nat-less [simp]:
(ordinal-of-nat x < ordinal-of-nat y) = (x < y)
<proof>
```

```
lemma ordinal-of-nat-le [simp]:
(ordinal-of-nat x  $\leq$  ordinal-of-nat y) = (x  $\leq$  y)
<proof>
```

```
lemma ordinal-of-nat-plus [simp]:
ordinal-of-nat x + ordinal-of-nat y = ordinal-of-nat (x + y)
<proof>
```

```
lemma ordinal-of-nat-times [simp]:
ordinal-of-nat x * ordinal-of-nat y = ordinal-of-nat (x * y)
<proof>
```

```
lemma ordinal-of-nat-exp [simp]:
ordinal-of-nat x ** ordinal-of-nat y = ordinal-of-nat (x ^ y)
<proof>
```

```
lemma oSuc-plus-ordinal-of-nat:
oSuc x + ordinal-of-nat n = oSuc (x + ordinal-of-nat n)
<proof>
```

```
lemma less-ordinal-of-nat:
(x < ordinal-of-nat n) = ( $\exists$  m. x = ordinal-of-nat m  $\wedge$  m < n)
<proof>
```

**lemma** *le-ordinal-of-nat*:  
 $(x \leq \text{ordinal-of-nat } n) = (\exists m. x = \text{ordinal-of-nat } m \wedge m \leq n)$   
 ⟨proof⟩

## 8.2 Omega, the least limit ordinal

**definition**  
*omega* :: ordinal ( $\omega$ ) **where**  
*omega* = *oLimit ordinal-of-nat*

**lemma** *less-omegaD*:  $x < \omega \implies \exists n. x = \text{ordinal-of-nat } n$   
 ⟨proof⟩

**lemma** *omega-leI*:  $\forall n. \text{ordinal-of-nat } n \leq x \implies \omega \leq x$   
 ⟨proof⟩

**lemma** *nat-le-omega* [*simp*]:  $\text{ordinal-of-nat } n \leq \omega$   
 ⟨proof⟩

**lemma** *nat-less-omega* [*simp*]:  $\text{ordinal-of-nat } n < \omega$   
 ⟨proof⟩

**lemma** *zero-less-omega* [*simp*]:  $0 < \omega$   
 ⟨proof⟩

**lemma** *limit-ordinal-omega*:  $\text{limit-ordinal } \omega$   
 ⟨proof⟩

**lemma** *Least-limit-ordinal*: (*LEAST*  $x. \text{limit-ordinal } x$ ) =  $\omega$   
 ⟨proof⟩

**lemma** *range f = range ordinal-of-nat*  $\implies \text{oLimit } f = \omega$   
 ⟨proof⟩

## 8.3 Arithmetic properties of $\omega$

**lemma** *oSuc-less-omega* [*simp*]:  $(\text{oSuc } x < \omega) = (x < \omega)$   
 ⟨proof⟩

**lemma** *oSuc-plus-omega* [*simp*]:  $\text{oSuc } x + \omega = x + \omega$   
 ⟨proof⟩

**lemma** *ordinal-of-nat-plus-omega* [*simp*]:  
 $\text{ordinal-of-nat } n + \omega = \omega$   
 ⟨proof⟩

**lemma** *ordinal-of-nat-times-omega* [*simp*]:  
 $0 < k \implies \text{ordinal-of-nat } k * \omega = \omega$   
 ⟨proof⟩

**lemma** *ordinal-plus-times-omega*:  $x + x * \omega = x * \omega$   
*<proof>*

**lemma** *ordinal-plus-absorb*:  $x * \omega \leq y \implies x + y = y$   
*<proof>*

**lemma** *ordinal-less-plusL*:  $y < x * \omega \implies y < x + y$   
*<proof>*

**lemma** *ordinal-plus-absorb-iff*:  $(x + y = y) = (x * \omega \leq y)$   
*<proof>*

**lemma** *ordinal-less-plusL-iff*:  $(y < x + y) = (y < x * \omega)$   
*<proof>*

## 8.4 Additive principal ordinals

**locale** *additive-principal* =  
  **fixes**  $a :: \text{ordinal}$   
  **assumes** *not-0*:  $0 < a$   
  **assumes** *sum-eq*:  $\bigwedge b. b < a \implies b + a = a$

**lemma** (**in** *additive-principal*) *sum-less*:  
 $\llbracket x < a; y < a \rrbracket \implies x + y < a$   
*<proof>*

**lemma** (**in** *additive-principal*) *times-nat-less*:  
 $x < a \implies x * \text{ordinal-of-nat } n < a$   
*<proof>*

**lemma** *not-additive-principal-0*:  $\neg \text{additive-principal } 0$   
*<proof>*

**lemma** *additive-principal-oSuc*:  
 $\text{additive-principal } (\text{oSuc } a) = (a = 0)$   
*<proof>*

**lemma** *additive-principal-intro2* [*rule-format*]:  
**assumes** *not-0*:  $0 < a$   
**shows**  $(\forall x < a. \forall y < a. x + y < a) \longrightarrow \text{additive-principal } a$   
*<proof>*

**lemma** *additive-principal-1*:  $\text{additive-principal } (\text{oSuc } 0)$   
*<proof>*

**lemma** *additive-principal-omega*:  $\text{additive-principal } \omega$   
*<proof>*



**lemma** *additive-principal-times-omega*:

$0 < x \implies \text{additive-principal } (x * \omega)$

*<proof>*

**lemma** *additive-principal-oLimit*:

$\forall n. \text{additive-principal } (f\ n) \implies \text{additive-principal } (oLimit\ f)$

*<proof>*

**lemma** *additive-principal-omega-exp*:  $\text{additive-principal } (\omega ** x)$

*<proof>*

**lemma** (*in additive-principal*) *omega-exp*:  $\exists x. a = \omega ** x$

*<proof>*

**lemma** *additive-principal-iff*:

$\text{additive-principal } a = (\exists x. a = \omega ** x)$

*<proof>*

**lemma** *absorb-omega-exp*:

$x < \omega ** a \implies x + \omega ** a = \omega ** a$

*<proof>*

**lemma** *absorb-omega-exp2*:  $a < b \implies \omega ** a + \omega ** b = \omega ** b$

*<proof>*

## 8.5 Cantor normal form

**lemma** *cnf-lemma*:  $x > 0 \implies x - \omega ** oLog\ \omega\ x < x$

*<proof>*

**primrec** *from-cnf* **where**

*from-cnf* [] = 0

| *from-cnf* (x # xs) =  $\omega ** x + \text{from-cnf } xs$

**function** *to-cnf* **where**

[*simp del*]:  $\text{to-cnf } x = (\text{if } x = 0 \text{ then } [] \text{ else}$

$oLog\ \omega\ x \# \text{to-cnf } (x - \omega ** oLog\ \omega\ x))$

*<proof>*

**termination** *<proof>*

**lemma** *to-cnf-0* [*simp*]:  $\text{to-cnf } 0 = []$

*<proof>*

**lemma** *to-cnf-not-0*:

$0 < x \implies \text{to-cnf } x = oLog\ \omega\ x \# \text{to-cnf } (x - \omega ** oLog\ \omega\ x)$

*<proof>*

**lemma** *to-cnf-eq-Cons*:  $\text{to-cnf } x = a \# list \implies a = oLog\ \omega\ x$

$\langle \text{proof} \rangle$

**lemma** *to-cnf-inverse*:  $\text{from-cnf } (\text{to-cnf } x) = x$

$\langle \text{proof} \rangle$

**primrec** *normalize-cnf* **where**

*normalize-cnf-Nil*:  $\text{normalize-cnf } [] = []$   
| *normalize-cnf-Cons*:  $\text{normalize-cnf } (x \# xs) =$   
    (*case xs of*  $[] \Rightarrow [x] \mid y \# ys \Rightarrow$   
    (*if*  $x < y$  *then*  $[]$  *else*  $[x]$ )  $\text{@ normalize-cnf } xs$ )

**lemma** *from-cnf-normalize-cnf*:  $\text{from-cnf } (\text{normalize-cnf } xs) = \text{from-cnf } xs$

$\langle \text{proof} \rangle$

**lemma** *normalize-cnf-to-cnf*:  $\text{normalize-cnf } (\text{to-cnf } x) = \text{to-cnf } x$

$\langle \text{proof} \rangle$

alternate form of CNF

**lemma** *cnf2-lemma*:

$0 < x \implies x \bmod \omega ** oLog \omega x < x$

$\langle \text{proof} \rangle$

**primrec** *from-cnf2* **where**

*from-cnf2*  $[] = 0$   
| *from-cnf2*  $(x \# xs) = \omega ** \text{fst } x * \text{ordinal-of-nat } (\text{snd } x) + \text{from-cnf2 } xs$

**function** *to-cnf2* **where**

[*simp del*]:  $\text{to-cnf2 } x = (\text{if } x = 0 \text{ then } [] \text{ else}$   
    ( $oLog \omega x, \text{inv ordinal-of-nat } (x \text{ div } (\omega ** oLog \omega x))$ )  
     $\# \text{to-cnf2 } (x \bmod (\omega ** oLog \omega x))$ )

$\langle \text{proof} \rangle$

**termination**  $\langle \text{proof} \rangle$

**lemma** *to-cnf2-0* [*simp*]:  $\text{to-cnf2 } 0 = []$

$\langle \text{proof} \rangle$

**lemma** *to-cnf2-not-0*:

$0 < x \implies \text{to-cnf2 } x =$   
    ( $oLog \omega x, \text{inv ordinal-of-nat } (x \text{ div } (\omega ** oLog \omega x))$ )  
     $\# \text{to-cnf2 } (x \bmod (\omega ** oLog \omega x))$

$\langle \text{proof} \rangle$

**lemma** *to-cnf2-eq-Cons*:  $\text{to-cnf2 } x = (a,b) \# \text{list} \implies a = oLog \omega x$

$\langle \text{proof} \rangle$

**lemma** *ordinal-of-nat-of-ordinal*:

$x < \omega \implies \text{ordinal-of-nat } (\text{inv ordinal-of-nat } x) = x$

$\langle \text{proof} \rangle$

**lemma** *to-cnf2-inverse*: *from-cnf2* (*to-cnf2*  $x$ ) =  $x$   
 ⟨*proof*⟩

**primrec** *is-normalized2* **where**

*is-normalized2-Nil*: *is-normalized2* [] = *True*  
 | *is-normalized2-Cons*: *is-normalized2* ( $x \# xs$ ) =  
 (case  $xs$  of []  $\Rightarrow$  *True* |  $y \# ys \Rightarrow$   $\text{fst } y < \text{fst } x \wedge \text{is-normalized2 } ys$ )

**lemma** *is-normalized2-to-cnf2*: *is-normalized2* (*to-cnf2*  $x$ )  
 ⟨*proof*⟩

## 8.6 Epsilon 0

**definition** *epsilon0* :: *ordinal* ( $\varepsilon_0$ ) **where**  
*epsilon0* = *oFix* (*op* \*\*  $\omega$ ) 0

**lemma** *less-omega-exp*:  $x < \varepsilon_0 \implies x < \omega ** x$   
 ⟨*proof*⟩

**lemma** *omega-exp-epsilon0*:  $\omega ** \varepsilon_0 = \varepsilon_0$   
 ⟨*proof*⟩

**lemma** *oLog-omega-less*:  $\llbracket 0 < x; x < \varepsilon_0 \rrbracket \implies \text{oLog } \omega x < x$   
 ⟨*proof*⟩

**end**

## 9 Veblen Hierarchies

**theory** *OrdinalVeblen*  
**imports** *OrdinalOmega*  
**begin**

### 9.1 Closed, unbounded sets

**locale** *normal-set* =  
**fixes**  $A$  :: *ordinal set*  
**assumes** *closed*:  $\bigwedge g. \forall n. g \ n \in A \implies \text{oLimit } g \in A$   
**and** *unbounded*:  $\bigwedge x. \exists y \in A. x < y$

**lemma** (**in** *normal-set*) *less-next*:  $x < (\text{LEAST } z. z \in A \wedge x < z)$   
 ⟨*proof*⟩

**lemma** (**in** *normal-set*) *mem-next*:  $(\text{LEAST } z. z \in A \wedge x < z) \in A$   
 ⟨*proof*⟩

**lemma** (**in** *normal*) *normal-set-range*: *normal-set* (*range*  $F$ )  
 ⟨*proof*⟩

**lemma** *oLimit-mem-INTER*:

$\llbracket \forall n. \text{normal-set } (A\ n); \forall n. A\ (\text{Suc } n) \subseteq A\ n;$   
 $\forall n. f\ n \in A\ n; \text{mono } f \rrbracket$   
 $\implies \text{oLimit } f \in (\bigcap n. A\ n)$   
 $\langle \text{proof} \rangle$

**lemma** *normal-set-INTER*:

$\llbracket \forall n. \text{normal-set } (A\ n); \forall n. A\ (\text{Suc } n) \subseteq A\ n \rrbracket \implies \text{normal-set } (\bigcap n. A\ n)$   
 $\langle \text{proof} \rangle$

## 9.2 Ordering functions

There is a one-to-one correspondence between closed, unbounded sets of ordinals and normal functions on ordinals.

**definition**

*ordering* :: (*ordinal set*)  $\Rightarrow$  (*ordinal*  $\Rightarrow$  *ordinal*) **where**  
*ordering*  $A = \text{ordinal-rec } (\text{LEAST } z. z \in A) (\lambda p\ x. \text{LEAST } z. z \in A \wedge x < z)$

**lemma** *ordering-0*:

*ordering*  $A\ 0 = (\text{LEAST } z. z \in A)$   
 $\langle \text{proof} \rangle$

**lemma** *ordering-oSuc*:

*ordering*  $A\ (\text{oSuc } x) = (\text{LEAST } z. z \in A \wedge \text{ordering } A\ x < z)$   
 $\langle \text{proof} \rangle$

**lemma** (**in** *normal-set*) *normal-ordering*: *normal* (*ordering*  $A$ )

$\langle \text{proof} \rangle$

**lemma** (**in** *normal-set*) *ordering-oLimit*:

*ordering*  $A\ (\text{oLimit } f) = \text{oLimit } (\lambda n. \text{ordering } A\ (f\ n))$   
 $\langle \text{proof} \rangle$

**lemma** (**in** *normal*) *ordering-range*: *ordering* (*range*  $F$ ) =  $F$

$\langle \text{proof} \rangle$

**lemma** (**in** *normal-set*) *ordering-mem*: *ordering*  $A\ x \in A$

$\langle \text{proof} \rangle$

**lemma** (**in** *normal-set*) *range-ordering-lemma*:

$\forall y. y \in A \longrightarrow y < \text{ordering } A\ x \longrightarrow y \in \text{range } (\text{ordering } A)$   
 $\langle \text{proof} \rangle$

**lemma** (**in** *normal-set*) *range-ordering*: *range* (*ordering*  $A$ ) =  $A$

$\langle \text{proof} \rangle$

**lemma** *ordering-INTER-0*:

$\llbracket \forall n. \text{normal-set } (A\ n); \forall n. A\ (\text{Suc } n) \subseteq A\ n \rrbracket$

$\implies \text{ordering } (\bigcap n. A n) 0 = \text{oLimit } (\lambda n. \text{ordering } (A n) 0)$   
 ⟨proof⟩

### 9.3 Critical ordinals

**definition**

*critical-set* :: ordinal set  $\Rightarrow$  ordinal  $\Rightarrow$  ordinal set **where**  
*critical-set* A =  
 ordinal-rec0 A ( $\lambda p x. x \cap \text{range } (\text{oDeriv } (\text{ordering } x))$ ) ( $\lambda f. \bigcap n. f n$ )

**lemma** *critical-set-0*:

*critical-set* A 0 = A  
 ⟨proof⟩

**lemma** *critical-set-oSuc-lemma*:

*critical-set* A (oSuc n) =  
*critical-set* A n  $\cap$  range (oDeriv (ordering (critical-set A n)))  
 ⟨proof⟩

**lemma** *omega-complete-INTER*:

*omega-complete* ( $\lambda x y. y \subseteq x$ ) (INTER UNIV)  
 ⟨proof⟩

**lemma** *critical-set-oLimit*:

*critical-set* A (oLimit f) = ( $\bigcap n. \text{critical-set } A (f n)$ )  
 ⟨proof⟩

**lemma** *critical-set-mono*:

$x \leq y \implies \text{critical-set } A y \subseteq \text{critical-set } A x$   
 ⟨proof⟩

**lemma** (in normal-set) *range-oDeriv-subset*:

range (oDeriv (ordering A))  $\subseteq$  A  
 ⟨proof⟩

**lemma** *normal-set-critical-set*:

normal-set A  $\implies$  normal-set (critical-set A x)  
 ⟨proof⟩

**lemma** *critical-set-oSuc*:

normal-set A  
 $\implies \text{critical-set } A (\text{oSuc } x) = \text{range } (\text{oDeriv } (\text{ordering } (\text{critical-set } A x)))$   
 ⟨proof⟩

### 9.4 Veblen hierarchy over a normal function

**definition**

*oVeblen* :: (ordinal  $\Rightarrow$  ordinal)  $\Rightarrow$  ordinal  $\Rightarrow$  ordinal  $\Rightarrow$  ordinal **where**  
*oVeblen* F = ( $\lambda x. \text{ordering } (\text{critical-set } (\text{range } F) x)$ )

**lemma** (in normal) *oVeblen-0*:  $oVeblen\ F\ 0 = F$   
⟨proof⟩

**lemma** (in normal) *oVeblen-oSuc*:  
 $oVeblen\ F\ (oSuc\ x) = oDeriv\ (oVeblen\ F\ x)$   
⟨proof⟩

**lemma** (in normal) *oVeblen-oLimit*:  
 $oVeblen\ F\ (oLimit\ f) = ordering\ (\bigcap n. range\ (oVeblen\ F\ (f\ n)))$   
⟨proof⟩

**lemma** (in normal) *normal-oVeblen*:  
 $normal\ (oVeblen\ F\ x)$   
⟨proof⟩

**lemma** (in normal) *continuous-oVeblen-0*:  
 $continuous\ (\lambda x. oVeblen\ F\ x\ 0)$   
⟨proof⟩

**lemma** (in normal) *oVeblen-oLimit-0*:  
 $oVeblen\ F\ (oLimit\ f)\ 0 = oLimit\ (\lambda n. oVeblen\ F\ (f\ n)\ 0)$   
⟨proof⟩

**lemma** (in normal) *normal-oVeblen-0*:  
 $0 < F\ 0 \implies normal\ (\lambda x. oVeblen\ F\ x\ 0)$   
⟨proof⟩

**lemma** (in normal) *range-oVeblen*:  
 $range\ (oVeblen\ F\ x) = critical-set\ (range\ F)\ x$   
⟨proof⟩

**lemma** (in normal) *range-oVeblen-subset*:  
 $x \leq y \implies range\ (oVeblen\ F\ y) \subseteq range\ (oVeblen\ F\ x)$   
⟨proof⟩

**lemma** (in normal) *oVeblen-fixed*:  
 $\forall x < y. \forall a. oVeblen\ F\ x\ (oVeblen\ F\ y\ a) = oVeblen\ F\ y\ a$   
⟨proof⟩

**lemma** (in normal) *critical-set-fixed*:  
 $0 < z \implies range\ (oVeblen\ F\ z) = \{x. \forall y < z. oVeblen\ F\ y\ x = x\}$   
⟨proof⟩

## 9.5 Veblen hierarchy over $\lambda x. 1 + x$

**lemma** *oDeriv-id*:  $oDeriv\ id = id$   
⟨proof⟩

**lemma** *oFix-plus*:  $oFix\ (\lambda x. a + x)\ 0 = a * \omega$

*<proof>*

**lemma** *oDeriv-plus*:  $oDeriv (op + a) = (op + (a * \omega))$

*<proof>*

**lemma** *oVeblen-1-plus*:  $oVeblen (op + 1) x = (op + (\omega ** x))$

*<proof>*

**end**