Countable Ordinals

Brian Huffman

August 16, 2018

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

1 Definition of Ordinals **3**
- 1.1 Preliminary datatype for ordinals . . . 3
- 1.2 Ordinal type . . . 5
- 1.3 Induction over ordinals . . . 6

2 Ordinal Induction **7**
- 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

3 Continuity **13**
- 3.1 Continuous functions . . . 13
- 3.2 Normal functions . . . 13

4 Recursive Definitions **14**
- 4.1 Partial orders . . . 15
- 4.2 Recursive definitions for *ordinal* ⇒ *ordinal* . . . 16

5 Ordinal Arithmetic **17**
5.1 Addition . . . 17
5.2 Subtraction . . . 18
5.3 Multiplication . . . 20
5.4 Exponentiation . . . 21

6 Inverse Functions **23**
6.1 Division . . . 24
6.2 Derived properties of division . . . 26
6.3 Logarithms . . . 26

7 Fixed-points **28**
7.1 Derivatives of ordinal functions . . . 29

8 Omega **30**
8.1 Embedding naturals in the ordinals . . . 30
8.2 Omega, the least limit ordinal . . . 31
8.3 Arithmetic properties of ω . . . 31
8.4 Additive principal ordinals . . . 32
8.5 Cantor normal form . . . 33
8.6 Epsilon 0 . . . 35

9 Veblen Hierarchies **35**
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 ⇒ ord0*

subterm ordering on ord0

definition
ord0-prec :: (*ord0* × *ord0*) *set* **where**
ord0-prec = (⋃*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* × *ord0*) *set* **where**
⟦∀ *a*. (*a*,*x*) ∈ *ord0-prec*$^+$ ⟶ (∃ *b*. (*b*,*y*) ∈ *ord0-prec*$^+$ ∧ (*a*,*b*) ∈ *ord0-leq*)⟧
⟹ (*x*,*y*) ∈ *ord0-leq*

lemma *ord0-leqI*:
⟦∀ *a*. (*a*,*x*) ∈ *ord0-prec*$^+$ ⟶ (*a*,*y*) ∈ *ord0-leq O ord0-prec*$^+$⟧
⟹ (*x*,*y*) ∈ *ord0-leq*
⟨*proof*⟩

lemma *ord0-leqD*:
⟦(*x*,*y*) ∈ *ord0-leq*; (*a*,*x*) ∈ *ord0-prec*$^+$⟧ ⟹ (*a*,*y*) ∈ *ord0-leq O ord0-prec*$^+$
⟨*proof*⟩

lemma *ord0-leq-refl*: (*x*, *x*) ∈ *ord0-leq*
⟨*proof*⟩

lemma *ord0-leq-trans*[*rule-format*]:
∀ *y*. (*x*,*y*) ∈ *ord0-leq* ⟶
(∀ *z*. (*y*,*z*) ∈ *ord0-leq* ⟶ (*x*,*z*) ∈ *ord0-leq*)
⟨*proof*⟩

lemma *wf-ord0-leq*: *wf* (*ord0-leq O ord0-prec*$^+$)
⟨*proof*⟩

ordering on ord0

instantiation *ord0* :: *ord*
begin

definition
ord0-less-def: $x < y \longleftrightarrow (x,y) \in$ *ord0-leq O ord0-prec*$^+$

definition
ord0-le-def: $x \leq y \longleftrightarrow (x,y) \in$ *ord0-leq*

instance ⟨*proof*⟩

end

lemma *ord0-order-refl*[*simp*]: $(x::ord0) \leq x$
⟨*proof*⟩

lemma *ord0-order-trans*: $[\![(x::ord0) \leq y;\ y \leq z]\!] \Longrightarrow x \leq z$
⟨*proof*⟩

lemma *ord0-wf*: *wf* $\{(x,y::ord0).\ x < y\}$
⟨*proof*⟩

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

lemma *ord0-leI*:
$[\![\forall\, a::ord0.\ a < x \longrightarrow a < y]\!] \Longrightarrow x \leq y$
⟨*proof*⟩

lemma *ord0-less-le-trans*:
$[\![(x::ord0) < y;\ y \leq z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-le-less-trans*:
$[\![(x::ord0) \leq y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *rev-ord0-le-less-trans*:
$[\![(y::ord0) < z;\ x \leq y]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-less-trans*:
$[\![(x::ord0) < y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-less-imp-le*: $(x::ord0) < y \Longrightarrow x \leq y$
⟨*proof*⟩

lemma *ord0-linear-lemma*:
fixes m :: *ord0* **and** n :: *ord0*
shows $m < n \vee n < m \vee (m \leq n \wedge n \leq m)$
⟨*proof*⟩

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*]:
$(\bigwedge z.\ x = Abs\text{-}ordinal\ (ord0rel\ ``\ \{z\}) \Longrightarrow P) \Longrightarrow P$
⟨*proof*⟩

instantiation *ordinal* :: *ord*
begin

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

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

instance ⟨*proof*⟩

end

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

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

lemma *equiv-ord0rel*: *equiv UNIV ord0rel*
⟨*proof*⟩

lemma *Abs-ordinal-eq*[*simp*]:
(*Abs-ordinal* (*ord0rel* `` $\{x\}$) = *Abs-ordinal* (*ord0rel* `` $\{y\}$))
= $(x \leq y \wedge y \leq x)$
⟨*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*: $x < \mathit{ord0\text{-}Lim}\ f \Longrightarrow \exists\, n.\ x \leq f\ n$
⟨*proof*⟩

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

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

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

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

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

order properties of 0 and strict limits

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

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

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

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

end

2 Ordinal Induction

theory *OrdinalInduct*
imports *OrdinalDef*
begin

2.1 Zero and successor ordinals

definition
oSuc :: *ordinal* ⇒ *ordinal* **where**
oSuc x = *oStrictLimit* (λ*n*. *x*)

lemma *less-oSuc*[*iff*]: *x* < *oSuc x*
⟨*proof*⟩

lemma *oSuc-leI*: *x* < *y* ⟹ *oSuc x* ≤ *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* ≤ *y*)
⟨*proof*⟩

lemma *ordinal-0-le* [*iff*]: *0* ≤ (*x*::*ordinal*)
⟨*proof*⟩

lemma *ordinal-not-less-0* [*iff*]: ¬ (*x*::*ordinal*) < *0*
⟨*proof*⟩

lemma *ordinal-le-0* [*iff*]: (*x* ≤ *0*) = (*x* = (*0*::*ordinal*))
⟨*proof*⟩

lemma *ordinal-neq-0* [*iff*]: (*x* ≠ *0*) = (*0* < (*x*::*ordinal*))
⟨*proof*⟩

lemma *ordinal-not-0-less* [*iff*]: (¬ *0* < *x*) = (*x* = (*0*::*ordinal*))
⟨*proof*⟩

lemma *oSuc-le-eq-less*: (*oSuc x* ≤ *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*:
$[\![x \leq oSuc\ y;\ x \leq y \Longrightarrow R;\ x = oSuc\ y \Longrightarrow R]\!] \Longrightarrow R$
⟨*proof*⟩

lemma *less-oSucE*:
$[\![x < oSuc\ y;\ x < y \Longrightarrow P;\ x = y \Longrightarrow P]\!] \Longrightarrow P$
⟨*proof*⟩

2.2 Strict monotonicity

locale *strict-mono* =
fixes f
assumes *strict-mono*: $A < B \Longrightarrow f\ A < f\ B$

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

lemma *strict-mono-natI*:
fixes $f :: nat \Rightarrow 'a::order$
shows $(\bigwedge n.\ f\ n < f\ (Suc\ n)) \Longrightarrow$ *strict-mono f*
⟨*proof*⟩

lemma *mono-natI*:
fixes $f :: nat \Rightarrow 'a::order$
shows $(\bigwedge n.\ f\ n \leq f\ (Suc\ n)) \Longrightarrow$ *mono f*
⟨*proof*⟩

lemma *strict-mono-mono*:
fixes $f :: 'a::order \Rightarrow 'b::order$
shows *strict-mono f* $\Longrightarrow$ *mono f*
⟨*proof*⟩

lemma *strict-mono-monoD*:

fixes $f :: 'a::order \Rightarrow 'b::order$
shows $[\![strict\text{-}mono\ f;\ A \leq B]\!] \Longrightarrow f\ A \leq f\ B$
⟨*proof*⟩

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

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

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

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*: $\forall\, n.\ f\ n \leq x \Longrightarrow oLimit\ f \leq x$
⟨*proof*⟩

lemma *le-oLimit* [*iff*]: $f\ n \leq oLimit\ f$
⟨*proof*⟩

lemma *le-oLimitI*: $x \leq f\ n \Longrightarrow x \leq oLimit\ f$
⟨*proof*⟩

lemma *less-oLimitI*: $x < f\ n \Longrightarrow x < oLimit\ f$
⟨*proof*⟩

lemma *less-oLimitD*: $x < oLimit\ f \Longrightarrow \exists\, n.\ x < f\ n$
⟨*proof*⟩

lemma *less-oLimitE*:
$[\![x < oLimit\ f;\ \bigwedge n.\ x < f\ n \Longrightarrow P]\!] \Longrightarrow P$
⟨*proof*⟩

lemma *le-oLimitE*:
$[\![x \leq oLimit\ f;\ \bigwedge n.\ x \leq f\ n \Longrightarrow R;\ x = oLimit\ f \Longrightarrow R]\!] \Longrightarrow R$
⟨*proof*⟩

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

⟨*proof*⟩

lemma *strict-mono-less-oLimit*:
strict-mono f ⟹ *f n* < *oLimit f*
⟨*proof*⟩

lemma *oLimit-eqI*:
⟦⋀*n*. ∃ *m*. *f n* ≤ *g m*; ⋀*n*. ∃ *m*. *g n* ≤ *f m*⟧ ⟹ *oLimit f* = *oLimit g*
⟨*proof*⟩

lemma *oLimit-Suc*:
f 0 < *oLimit f* ⟹ *oLimit* (λ*n*. *f* (*Suc n*)) = *oLimit f*
⟨*proof*⟩

lemma *oLimit-shift*:
∀ *n*. *f n* < *oLimit f* ⟹ *oLimit* (λ*n*. *f* (*n* + *k*)) = *oLimit f*
⟨*proof*⟩

lemma *oLimit-shift-mono*:
mono f ⟹ *oLimit* (λ*n*. *f* (*n* + *k*)) = *oLimit f*
⟨*proof*⟩

limit ordinal predicate

definition
limit-ordinal :: *ordinal* ⇒ *bool* **where**
limit-ordinal x ⟷ (*x* ≠ 0) ∧ (∀ *y*. *x* ≠ *oSuc y*)

lemma *limit-ordinal-not-0* [*simp*]: ¬ *limit-ordinal* 0
⟨*proof*⟩

lemma *zero-less-limit-ordinal* [*simp*]: *limit-ordinal x* ⟹ 0 < *x*
⟨*proof*⟩

lemma *limit-ordinal-not-oSuc* [*simp*]: ¬ *limit-ordinal* (*oSuc p*)
⟨*proof*⟩

lemma *oSuc-less-limit-ordinal*:
limit-ordinal x ⟹ (*oSuc w* < *x*) = (*w* < *x*)
⟨*proof*⟩

lemma *limit-ordinal-oLimitI*:
∀ *n*. *f n* < *oLimit f* ⟹ *limit-ordinal* (*oLimit f*)
⟨*proof*⟩

lemma *strict-mono-limit-ordinal*:
strict-mono f ⟹ *limit-ordinal* (*oLimit f*)
⟨*proof*⟩

lemma *limit-ordinalI*:

$[\![0 < z; \forall x{<}z.\ oSuc\ x < z]\!] \Longrightarrow limit\text{-}ordinal\ z$
⟨*proof*⟩

2.3.1 Making strict monotonic sequences

primrec *make-mono* :: $(nat \Rightarrow ordinal) \Rightarrow nat \Rightarrow nat$
where
  $make\text{-}mono\ f\ 0 \quad = 0$
| $make\text{-}mono\ f\ (Suc\ n) = (LEAST\ x.\ f\ (make\text{-}mono\ f\ n) < f\ x)$

lemma *f-make-mono-less*:
$\forall\, n.\ f\ n < oLimit\ f \Longrightarrow f\ (make\text{-}mono\ f\ n) < f\ (make\text{-}mono\ f\ (Suc\ n))$
⟨*proof*⟩

lemma *strict-mono-f-make-mono*:
$\forall\, n.\ f\ n < oLimit\ f \Longrightarrow strict\text{-}mono\ (\lambda n.\ f\ (make\text{-}mono\ f\ n))$
⟨*proof*⟩

lemma *le-f-make-mono*:
$[\![\forall\, n.\ f\ n < oLimit\ f;\ m \leq make\text{-}mono\ f\ n]\!] \Longrightarrow f\ m \leq f\ (make\text{-}mono\ f\ n)$
⟨*proof*⟩

lemma *make-mono-less*:
$\forall\, n.\ f\ n < oLimit\ f \Longrightarrow make\text{-}mono\ f\ n < make\text{-}mono\ f\ (Suc\ n)$
⟨*proof*⟩

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

lemma *oLimit-make-mono-eq*:
$\forall\, n.\ f\ n < oLimit\ f \Longrightarrow oLimit\ (\lambda n.\ f\ (make\text{-}mono\ f\ n)) = oLimit\ f$
⟨*proof*⟩

2.4 Induction principle for ordinals

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

lemma *oLimit-induct*:
assumes *zero*: $P\ 0$
  and *suc*: $\bigwedge x.\ P\ x \Longrightarrow P\ (oSuc\ x)$
  and *lim*: $\bigwedge f.\ [\![strict\text{-}mono\ f;\ \forall\, n.\ P\ (f\ n)]\!] \Longrightarrow P\ (oLimit\ f)$
shows $P\ a$
⟨*proof*⟩

lemma *ordinal-cases*:
assumes *zero*: $a = 0 \Longrightarrow P$
  and *suc*: $\bigwedge x.\ a = oSuc\ x \Longrightarrow P$
  and *lim*: $\bigwedge f.\ [\![strict\text{-}mono\ f;\ a = oLimit\ f]\!] \Longrightarrow P$
shows P
⟨*proof*⟩

end

3 Continuity

theory *OrdinalCont*
imports *OrdinalInduct*
begin

3.1 Continuous functions

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

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

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

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

end

4 Recursive Definitions

theory *OrdinalRec*
imports *OrdinalCont*
begin

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

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

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

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

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

lemma *ordinal-rec0-0*:
ordinal-rec0 z s l 0 $= z$
⟨*proof*⟩

lemma *ordinal-rec0-oSuc*:
ordinal-rec0 z s l $(oSuc\ x) = s\ x\ (ordinal\text{-}rec0\ z\ s\ l\ x)$
⟨*proof*⟩

lemma *limit-ordinal-not-0*: *limit-ordinal* $x \Longrightarrow x \neq 0$
⟨*proof*⟩

lemma *limit-ordinal-not-oSuc*: *limit-ordinal* $x \Longrightarrow x \neq oSuc\ p$
⟨*proof*⟩

lemma *ordinal-rec0-limit-ordinal*:
limit-ordinal $x \Longrightarrow ordinal\text{-}rec0\ z\ s\ l\ x =$
$(THE\ y.\ \forall f.\ (\forall n.\ f\ n < oLimit\ f) \wedge oLimit\ f = x \longrightarrow$
$l\ (\lambda n.\ ordinal\text{-}rec0\ z\ s\ l\ (f\ n)) = y)$
⟨*proof*⟩

4.1 Partial orders

locale *porder* =
fixes *le* :: $'a \Rightarrow 'a \Rightarrow bool$ (**infixl** $<<$ 55)
assumes *po-refl*: $\bigwedge x.\ x << x$
and *po-trans*: $\bigwedge x\ y\ z.\ [\![x << y;\ y << z]\!] \Longrightarrow x << z$
and *po-antisym*: $\bigwedge x\ y.\ [\![x << y;\ y << x]\!] \Longrightarrow x = y$

lemma *porder-order*: *porder* $((\leq) :: 'a{::}order \Rightarrow 'a \Rightarrow bool)$
⟨*proof*⟩

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

locale *omega-complete* = *porder* +
fixes *lub* :: $(nat \Rightarrow 'a) \Rightarrow 'a$
assumes *is-ub-lub*: $\bigwedge f\ n.\ f\ n << lub\ f$
assumes *is-lub-lub*: $\bigwedge f\ x.\ \forall n.\ f\ n << x \Longrightarrow lub\ f << x$

lemma (**in** *omega-complete*) *lub-cong-lemma*:
$[\![\forall n.\ f\ n < oLimit\ f;\ \forall m.\ g\ m < oLimit\ g;\ oLimit\ f \leq oLimit\ g;$
$\forall y{<}oLimit\ g.\ \forall x{\leq}y.\ F\ x << F\ y]\!]$
$\Longrightarrow lub\ (\lambda n.\ F\ (f\ n)) << lub\ (\lambda n.\ F\ (g\ n))$
⟨*proof*⟩

lemma (**in** *omega-complete*) *lub-cong*:
$[\![\forall\, n.\ f\ n < oLimit\ f;\ \forall\, m.\ g\ m < oLimit\ g;\ oLimit\ f = oLimit\ g;$
$\forall\, y{<}oLimit\ g.\ \forall\, x{\leq}y.\ F\ x << F\ y]\!]$
$\Longrightarrow lub\ (\lambda n.\ F\ (f\ n)) = lub\ (\lambda n.\ F\ (g\ n))$
⟨*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 $x \leq y \Longrightarrow$ *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* ⇒ *ordinal*

definition
ordinal-rec ::
[*ordinal*, *ordinal* ⇒ *ordinal* ⇒ *ordinal*, *ordinal*] ⇒ *ordinal* **where**
ordinal-rec z s = *ordinal-rec0 z s oLimit*

lemma *omega-complete-oLimit*: *omega-complete* (≤) *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
$(+) = (\lambda x.\ ordinal\text{-}rec\ x\ (\lambda p.\ oSuc))$

instance ⟨*proof*⟩

end

lemma *normal-plus*: *normal* $((+)\ x)$
⟨*proof*⟩

lemma *ordinal-plus-0* [*simp*]: $x + 0 = (x{::}ordinal)$
⟨*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*]: $0 + x = (x{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-assoc*:
$(x + y) + z = x + (y + z{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-monoL* [*rule-format*]:
$\forall\, x\ x'.\ x \leq x' \longrightarrow x + y \leq x' + (y{::}ordinal)$
⟨*proof*⟩

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

lemma *ordinal-plus-mono*:
$[\![x \leq x';\ y \leq y']\!] \Longrightarrow x + y \leq x' + (y'{::}ordinal)$
⟨*proof*⟩

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

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

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

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

lemma *ordinal-plus-left-cancel* [*simp*]:
$(w + x = w + y) = (x = (y{::}ordinal))$
⟨*proof*⟩

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

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

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

lemma *not-inject*: $(\neg\ P) = (\neg\ Q) \Longrightarrow P = Q$
⟨*proof*⟩

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

5.2 Subtraction

instantiation *ordinal* :: *minus*

begin

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

instance ⟨*proof*⟩

end

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

lemma *ordinal-0-minus* [*simp*]: $0 - x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-oSuc-minus* [*simp*]: $y \leq x \Longrightarrow oSuc\ x - y = oSuc\ (x - y)$
⟨*proof*⟩

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

lemma *ordinal-minus-0* [*simp*]: $x - 0 = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-oSuc-minus2*: $x < y \Longrightarrow oSuc\ x - y = x - y$
⟨*proof*⟩

lemma *ordinal-minus-eq-0* [*rule-format*, *simp*]:
$x \leq y \longrightarrow x - y = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-minus1* [*simp*]: $(x + y) - x = (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-minus2* [*simp*]: $x \leq y \Longrightarrow x + (y - x) = (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-minusI*: $x = y + z \Longrightarrow x - y = (z::ordinal)$
⟨*proof*⟩

lemma *ordinal-minus-less-eq* [*simp*]:
$(y::ordinal) \leq x \Longrightarrow (x - y < z) = (x < y + z)$
⟨*proof*⟩

lemma *ordinal-minus-le-eq* [*simp*]:
$(x - y \leq z) = (x \leq y + (z::ordinal))$
⟨*proof*⟩

lemma *ordinal-minus-monoL*: $x \leq y \Longrightarrow x - z \leq y - (z::ordinal)$
⟨*proof*⟩

lemma *ordinal-minus-monoR*: $x \leq y \Longrightarrow z - y \leq z - (x::ordinal)$
⟨*proof*⟩

5.3 Multiplication

instantiation *ordinal* :: *times*
begin

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

instance ⟨*proof*⟩

end

lemma *continuous-times*: *continuous* $((*)\ x)$
⟨*proof*⟩

lemma *normal-times*: $0 < x \Longrightarrow normal\ ((*)\ x)$
⟨*proof*⟩

lemma *ordinal-times-0* [*simp*]: $x * 0 = (0::ordinal)$
⟨*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*]: $0 * x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-1-times* [*simp*]: $oSuc\ 0 * x = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-1* [*simp*]: $x * oSuc\ 0 = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-distrib*:
$x * (y + z) = (x * y) + (x * z::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-assoc*:
$(x * y::ordinal) * z = x * (y * z)$
⟨*proof*⟩

lemma *ordinal-times-monoL* [*rule-format*]:
$\forall x\ x'.\ x \leq x' \longrightarrow x * y \leq x' * (y::ordinal)$
⟨*proof*⟩

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

lemma *ordinal-times-mono*:
$[\![x \leq x';\ y \leq y']\!] \Longrightarrow x * y \leq x' * (y'::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-strict-monoR*:
$[\![y < y';\ 0 < x]\!] \Longrightarrow x * y < x * (y'::ordinal)$
⟨*proof*⟩

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

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

lemma *ordinal-less-timesR*: $[\![0 < x;\ oSuc\ 0 < y]\!] \Longrightarrow x < x * (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-left-cancel* [*simp*]:
$0 < w \Longrightarrow (w * x = w * y) = (x = (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-times-left-cancel-le* [*simp*]:
$0 < w \Longrightarrow (w * x \leq w * y) = (x \leq (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-times-left-cancel-less* [*simp*]:
$0 < w \Longrightarrow (w * x < w * y) = (x < (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-times-eq-0*:
$((x::ordinal) * y = 0) = (x = 0 \vee y = 0)$
⟨*proof*⟩

lemma *ordinal-times-not-0* [*simp*]:
$((0::ordinal) < x * y) = (0 < x \wedge 0 < y)$
⟨*proof*⟩

5.4 Exponentiation

definition
exp-ordinal :: [*ordinal*, *ordinal*] ⇒ *ordinal* (**infixr** ∗∗ 75) **where**

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

lemma *continuous-exp*: $0 < x \Longrightarrow \mathit{continuous}\ ((\ **\ )\ x)$
⟨*proof*⟩

lemma *ordinal-exp-0* [*simp*]: $x ** 0 = (1::\mathit{ordinal})$
⟨*proof*⟩

lemma *ordinal-exp-oSuc* [*simp*]: $x ** \mathit{oSuc}\ y = (x ** y) * x$
⟨*proof*⟩

lemma *ordinal-exp-oLimit* [*simp*]:
$0 < x \Longrightarrow x ** \mathit{oLimit}\ f = \mathit{oLimit}\ (\lambda n.\ x ** f\ n)$
⟨*proof*⟩

lemma *ordinal-0-exp* [*simp*]: $0 ** x = (\mathit{if}\ x = 0\ \mathit{then}\ 1\ \mathit{else}\ 0)$
⟨*proof*⟩

lemma *ordinal-1-exp* [*simp*]: $\mathit{oSuc}\ 0 ** x = \mathit{oSuc}\ 0$
⟨*proof*⟩

lemma *ordinal-exp-1* [*simp*]: $x ** \mathit{oSuc}\ 0 = x$
⟨*proof*⟩

lemma *ordinal-exp-distrib*:
$x ** (y + z) = (x ** y) * (x ** (z::\mathit{ordinal}))$
⟨*proof*⟩

lemma *ordinal-exp-not-0* [*simp*]: $(0 < x ** y) = (0 < x \vee y = 0)$
⟨*proof*⟩

lemma *ordinal-exp-eq-0* [*simp*]: $(x ** y = 0) = (x = 0 \wedge 0 < y)$
⟨*proof*⟩

lemma *ordinal-exp-assoc*:
$(x ** y) ** z = x ** (y * z)$
⟨*proof*⟩

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

lemma *normal-exp*: $\mathit{oSuc}\ 0 < x \Longrightarrow \mathit{normal}\ ((\ **\ )\ x)$
⟨*proof*⟩

lemma *ordinal-exp-monoR*:
$[\![0 < x;\ y \leq y']\!] \Longrightarrow x ** y \leq x ** (y'::\mathit{ordinal})$
⟨*proof*⟩

lemma *ordinal-exp-mono*:
$[\![0 < x'; x \leq x'; y \leq y']\!] \Longrightarrow x ** y \leq x' ** (y'::ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-strict-monoR*:
$[\![oSuc\ 0 < x; y < y']\!] \Longrightarrow x ** y < x ** (y'::ordinal)$
⟨*proof*⟩

lemma *ordinal-le-expR* [*simp*]: $0 < y \Longrightarrow x \leq x ** (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x = w ** y) = (x = y)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel-le* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x \leq w ** y) = (x \leq y)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel-less* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x < w ** y) = (x < y)$
⟨*proof*⟩

end

6 Inverse Functions

theory *OrdinalInverse*
imports *OrdinalArith*
begin

lemma (**in** *normal*) *oInv-ex*:
$F\ 0 \leq a \Longrightarrow \exists q.\ F\ q \leq a \wedge a < F\ (oSuc\ q)$
⟨*proof*⟩

lemma *oInv-uniq*:
$[\![mono\ (F::ordinal \Rightarrow ordinal);$
$F\ x \leq a \wedge a < F\ (oSuc\ x); F\ y \leq a \wedge a < F\ (oSuc\ y)]\!]$
$\Longrightarrow x = y$
⟨*proof*⟩

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

lemma (**in** *normal*) *oInv-bounds*:
$F\ 0 \leq a \Longrightarrow F\ (oInv\ F\ a) \leq a \wedge a < F\ (oSuc\ (oInv\ F\ a))$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-bound1*:
$F\ 0 \leq a \Longrightarrow F\ (oInv\ F\ a) \leq a$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-bound2*:
$a < F\ (oSuc\ (oInv\ F\ a))$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-equality*:
$[\![F\ x \leq a;\ a < F\ (oSuc\ x)]\!] \Longrightarrow oInv\ F\ a = x$
⟨*proof*⟩

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

lemma (**in** *normal*) *oInv-equality'*: $a = F\ x \Longrightarrow oInv\ F\ a = x$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-eq-0*: $a \leq F\ 0 \Longrightarrow oInv\ F\ a = 0$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-less*:
$[\![F\ 0 \leq a;\ a < F\ z]\!] \Longrightarrow oInv\ F\ a < z$
⟨*proof*⟩

lemma (**in** *normal*) *le-oInv*:
$F\ z \leq a \Longrightarrow z \leq oInv\ F\ a$
⟨*proof*⟩

lemma (**in** *normal*) *less-oInvD*:
$x < oInv\ F\ a \Longrightarrow F\ (oSuc\ x) \leq a$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-le*:
$a < F\ (oSuc\ x) \Longrightarrow oInv\ F\ a \leq x$
⟨*proof*⟩

lemma (**in** *normal*) *mono-oInv*: $mono\ (oInv\ F)$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-decreasing*:
$F\ 0 \leq x \Longrightarrow oInv\ F\ x \leq x$
⟨*proof*⟩

6.1 Division

instantiation *ordinal* :: *modulo*
begin

definition
div-ordinal-def:
$x\ div\ y = (if\ 0 < y\ then\ oInv\ ((\ *\ )\ y)\ x\ else\ 0)$

definition
mod-ordinal-def:
$x\ mod\ y = ((x::ordinal) - y * (x\ div\ y))$

instance ⟨*proof*⟩

end

lemma *ordinal-divI*: $[\![x = y * q + r;\ r < y]\!] \Longrightarrow x\ div\ y = (q::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-div-le*: $y * (x\ div\ y) \leq (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-less-times-div-plus*:
$0 < y \Longrightarrow x < y * (x\ div\ y) + (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-modI*: $[\![x = y * q + r;\ r < y]\!] \Longrightarrow x\ mod\ y = (r::ordinal)$
⟨*proof*⟩

lemma *ordinal-mod-less*: $0 < y \Longrightarrow x\ mod\ y < (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-plus-mod*: $y * (x\ div\ y) + (x\ mod\ y) = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-less*: $x < y * z \Longrightarrow x\ div\ y < (z::ordinal)$
⟨*proof*⟩

lemma *ordinal-le-div*: $[\![0 < y;\ y * z \leq x]\!] \Longrightarrow (z::ordinal) \leq x\ div\ y$
⟨*proof*⟩

lemma *ordinal-mono-div*: $mono\ (\lambda x.\ x\ div\ y::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-monoL*: $x \leq x' \Longrightarrow x\ div\ y \leq x'\ div\ (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-decreasing*: $(x::ordinal)\ div\ y \leq x$
⟨*proof*⟩

lemma *ordinal-div-0*: $x\ div\ 0 = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-mod-0*: $x\ mod\ 0 = (x::ordinal)$
⟨*proof*⟩

6.2 Derived properties of division

lemma *ordinal-div-1* [*simp*]: $x\ div\ oSuc\ 0 = x$
⟨*proof*⟩

lemma *ordinal-mod-1* [*simp*]: $x\ mod\ oSuc\ 0 = 0$
⟨*proof*⟩

lemma *ordinal-div-self* [*simp*]: $0 < x \Longrightarrow x\ div\ x = (1::ordinal)$
⟨*proof*⟩

lemma *ordinal-mod-self* [*simp*]: $x\ mod\ x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-greater* [*simp*]: $x < y \Longrightarrow x\ div\ y = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-mod-greater* [*simp*]: $x < y \Longrightarrow x\ mod\ y = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-0-div* [*simp*]: $0\ div\ x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-0-mod* [*simp*]: $0\ mod\ x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-1-dvd* [*simp*]: $oSuc\ 0\ dvd\ x$
⟨*proof*⟩

lemma *ordinal-dvd-mod*: $y\ dvd\ x = (x\ mod\ y = (0::ordinal))$
⟨*proof*⟩

lemma *ordinal-dvd-times-div*:
$y\ dvd\ x \Longrightarrow y * (x\ div\ y) = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-dvd-oLimit*: $\forall\, n.\ x\ dvd\ f\ n \Longrightarrow x\ dvd\ oLimit\ f$
⟨*proof*⟩

6.3 Logarithms

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

lemma *ordinal-oLogI*:

$[\![b ** y \leq x; x < b ** y * b]\!] \Longrightarrow oLog\ b\ x = y$
⟨*proof*⟩

lemma *ordinal-exp-oLog-le*:
$[\![0 < x; oSuc\ 0 < b]\!] \Longrightarrow b ** (oLog\ b\ x) \leq x$
⟨*proof*⟩

lemma *ordinal-less-exp-oLog*:
$oSuc\ 0 < b \Longrightarrow x < b ** (oLog\ b\ x) * b$
⟨*proof*⟩

lemma *ordinal-oLog-less*:
$[\![0 < x; oSuc\ 0 < b; x < b ** y]\!] \Longrightarrow oLog\ b\ x < y$
⟨*proof*⟩

lemma *ordinal-le-oLog*:
$[\![oSuc\ 0 < b; b ** y \leq x]\!] \Longrightarrow y \leq oLog\ b\ x$
⟨*proof*⟩

lemma *ordinal-oLogI2*:
$[\![oSuc\ 0 < b; x = b ** y * q + r; 0 < q; q < b; r < b ** y]\!] \Longrightarrow oLog\ b\ x = y$
⟨*proof*⟩

lemma *ordinal-div-exp-oLog-less*:
$oSuc\ 0 < b \Longrightarrow x\ div\ (b ** oLog\ b\ x) < b$
⟨*proof*⟩

lemma *ordinal-oLog-base-0*: $oLog\ 0\ x = 0$
⟨*proof*⟩

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

lemma *ordinal-oLog-0*: $oLog\ b\ 0 = 0$
⟨*proof*⟩

lemma *ordinal-oLog-exp*: $oSuc\ 0 < b \Longrightarrow oLog\ b\ (b ** x) = x$
⟨*proof*⟩

lemma *ordinal-oLog-self*: $oSuc\ 0 < b \Longrightarrow oLog\ b\ b = oSuc\ 0$
⟨*proof*⟩

lemma *ordinal-mono-oLog*: $mono\ (oLog\ b)$
⟨*proof*⟩

lemma *ordinal-oLog-monoR*: $x \leq y \Longrightarrow oLog\ b\ x \leq oLog\ b\ y$
⟨*proof*⟩

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

⟨*proof*⟩

end

7 Fixed-points

theory *OrdinalFix*
imports *OrdinalInverse*
begin

primrec *iter* :: *nat* ⇒ (′*a* ⇒ ′*a*) ⇒ (′*a* ⇒ ′*a*)
where
  iter 0 *F x* = *x*
| *iter* (*Suc n*) *F x* = *F* (*iter n F x*)

definition
  oFix :: (*ordinal* ⇒ *ordinal*) ⇒ *ordinal* ⇒ *ordinal* **where**
  oFix F a = *oLimit* (λ*n*. *iter n F a*)

lemma *oFix-fixed*:
⟦*continuous F*; *a* ≤ *F a*⟧ ⟹ *F* (*oFix F a*) = *oFix F a*
⟨*proof*⟩

lemma *oFix-least*:
⟦*mono F*; *F x* = *x*; *a* ≤ *x*⟧ ⟹ *oFix F a* ≤ *x*
⟨*proof*⟩

lemma *mono-oFix*: *mono F* ⟹ *mono* (*oFix F*)
⟨*proof*⟩

lemma *less-oFixD*:
⟦*x* < *oFix F a*; *mono F*; *F x* = *x*⟧ ⟹ *x* < *a*
⟨*proof*⟩

lemma *less-oFixI*: *a* < *F a* ⟹ *a* < *oFix F a*
⟨*proof*⟩

lemma *le-oFix*: *a* ≤ *oFix F a*
⟨*proof*⟩

lemma *le-oFix1*: *F a* ≤ *oFix F a*
⟨*proof*⟩

lemma *less-oFix-0D*:
⟦*x* < *oFix F 0*; *mono F*⟧ ⟹ *x* < *F x*
⟨*proof*⟩

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

lemma *oFix-eq-self*: $F\ a = a \Longrightarrow oFix\ F\ a = a$
⟨*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**
$oDeriv\ F = ordinal\text{-}rec\ (oFix\ F\ 0)\ (\lambda p\ x.\ oFix\ F\ (oSuc\ x))$

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*:
$normal\ F \Longrightarrow F\ (oDeriv\ F\ n) = oDeriv\ F\ n$
⟨*proof*⟩

lemma *oDeriv-fixedD*:
$[\![oDeriv\ F\ x = x;\ normal\ F]\!] \Longrightarrow F\ x = x$
⟨*proof*⟩

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

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

lemma *oDeriv-total*:
$[\![normal\ F;\ F\ x = x]\!] \Longrightarrow \exists\, n.\ x = oDeriv\ F\ n$
⟨*proof*⟩

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

end

8 Omega

theory *OrdinalOmega*
imports *OrdinalFix*
begin

8.1 Embedding naturals in the ordinals

primrec *ordinal-of-nat* :: *nat* ⇒ *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*: ¬ *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* ≤ *ordinal-of-nat y*) = (*x* ≤ *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*) = (∃ *m*. *x* = *ordinal-of-nat m* ∧ *m* < *n*)
 ⟨*proof*⟩

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

8.2 Omega, the least limit ordinal

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

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

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

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

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

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

lemma *limit-ordinal-omega*: *limit-ordinal* ω
⟨*proof*⟩

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

lemma *range f* = *range ordinal-of-nat* $\Longrightarrow$ *oLimit f* = ω
⟨*proof*⟩

8.3 Arithmetic properties of ω

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

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

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

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

lemma *ordinal-plus-times-omega*: $x + x * \omega = x * \omega$
⟨*proof*⟩

lemma *ordinal-plus-absorb*: $x * \omega \leq y \Longrightarrow x + y = y$
⟨*proof*⟩

lemma *ordinal-less-plusL*: $y < x * \omega \Longrightarrow y < x + y$
⟨*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 :: *ordinal*
  assumes *not-0*: $0 < a$
  assumes *sum-eq*: $\bigwedge b.\ b < a \Longrightarrow b + a = a$

lemma (**in** *additive-principal*) *sum-less*:
$[\![x < a;\ y < a]\!] \Longrightarrow x + y < a$
⟨*proof*⟩

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

lemma *not-additive-principal-0*: ¬ *additive-principal 0*
⟨*proof*⟩

lemma *additive-principal-oSuc*:
additive-principal (*oSuc a*) = (*a* = *0*)
⟨*proof*⟩

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

lemma *additive-principal-1*: *additive-principal* (*oSuc 0*)
⟨*proof*⟩

lemma *additive-principal-omega*: *additive-principal* ω
⟨*proof*⟩

lemma *additive-principal-times-omega*:
$0 < x \Longrightarrow$ *additive-principal* $(x * \omega)$
⟨*proof*⟩

lemma *additive-principal-oLimit*:
$\forall\, n.$ *additive-principal* $(f\ n) \Longrightarrow$ *additive-principal* $(oLimit\ f)$
⟨*proof*⟩

lemma *additive-principal-omega-exp*: *additive-principal* $(\omega\ **\ x)$
⟨*proof*⟩

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

lemma *additive-principal-iff*:
additive-principal $a = (\exists\, x.\ a = \omega\ **\ x)$
⟨*proof*⟩

lemma *absorb-omega-exp*:
$x < \omega\ **\ a \Longrightarrow x + \omega\ **\ a = \omega\ **\ a$
⟨*proof*⟩

lemma *absorb-omega-exp2*: $a < b \Longrightarrow \omega\ **\ a + \omega\ **\ b = \omega\ **\ b$
⟨*proof*⟩

8.5 Cantor normal form

lemma *cnf-lemma*: $x > 0 \Longrightarrow x - \omega\ **\ oLog\ \omega\ x < x$
⟨*proof*⟩

primrec *from-cnf* **where**
from-cnf $[] \quad = 0$
| *from-cnf* $(x\ \#\ xs) = \omega\ **\ x + from\text{-}cnf\ xs$

function *to-cnf* **where**
[*simp del*]: *to-cnf* $x =$ (*if* $x = 0$ *then* $[]$ *else*
$oLog\ \omega\ x\ \#$ *to-cnf* $(x - \omega\ **\ oLog\ \omega\ x)$)
⟨*proof*⟩

termination ⟨*proof*⟩

lemma *to-cnf-0* [*simp*]: *to-cnf* $0 = []$
⟨*proof*⟩

lemma *to-cnf-not-0*:
$0 < x \Longrightarrow$ *to-cnf* $x = oLog\ \omega\ x\ \#$ *to-cnf* $(x - \omega\ **\ oLog\ \omega\ x)$
⟨*proof*⟩

lemma *to-cnf-eq-Cons*: *to-cnf* $x = a\ \#\ list \Longrightarrow a = oLog\ \omega\ x$

⟨*proof*⟩

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

primrec *normalize-cnf* **where**
normalize-cnf-Nil: *normalize-cnf* [] = []
| *normalize-cnf-Cons*: *normalize-cnf* (*x* # *xs*) =
(*case xs of* [] ⇒ [*x*] | *y* # *ys* ⇒
(*if x* < *y then* [] *else* [*x*]) @ *normalize-cnf xs*)

lemma *from-cnf-normalize-cnf*: *from-cnf* (*normalize-cnf xs*) = *from-cnf xs*
⟨*proof*⟩

lemma *normalize-cnf-to-cnf*: *normalize-cnf* (*to-cnf x*) = *to-cnf x*
⟨*proof*⟩

alternate form of CNF

lemma *cnf2-lemma*:
0 < *x* ⟹ *x mod* ω ** *oLog* ω *x* < *x*
⟨*proof*⟩

primrec *from-cnf2* **where**
from-cnf2 [] = *0*
| *from-cnf2* (*x* # *xs*) = ω ** *fst x* * *ordinal-of-nat* (*snd x*) + *from-cnf2 xs*

function *to-cnf2* **where**
[*simp del*]: *to-cnf2 x* = (*if x* = *0 then* [] *else*
(*oLog* ω *x*, *inv ordinal-of-nat* (*x div* (ω ** *oLog* ω *x*)))
to-cnf2 (*x mod* (ω ** *oLog* ω *x*)))
⟨*proof*⟩

termination ⟨*proof*⟩

lemma *to-cnf2-0* [*simp*]: *to-cnf2 0* = []
⟨*proof*⟩

lemma *to-cnf2-not-0*:
0 < *x* ⟹ *to-cnf2 x* =
(*oLog* ω *x*, *inv ordinal-of-nat* (*x div* (ω ** *oLog* ω *x*)))
to-cnf2 (*x mod* (ω ** *oLog* ω *x*))
⟨*proof*⟩

lemma *to-cnf2-eq-Cons*: *to-cnf2 x* = (*a*,*b*) # *list* ⟹ *a* = *oLog* ω *x*
⟨*proof*⟩

lemma *ordinal-of-nat-of-ordinal*:
x < ω ⟹ *ordinal-of-nat* (*inv ordinal-of-nat x*) = *x*
⟨*proof*⟩

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* [] ⇒ *True* | *y* # *ys* ⇒ *fst y* < *fst x* ∧ *is-normalized2 xs*)

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

8.6 Epsilon 0

definition *epsilon0* :: *ordinal* (*ε*$_0$) **where**
epsilon0 = *oFix* ((**)) *ω*) *0*

lemma *less-omega-exp*: *x* < *ε*$_0$ ⟹ *x* < *ω* ** *x*
⟨*proof*⟩

lemma *omega-exp-epsilon0*: *ω* ** *ε*$_0$ = *ε*$_0$
⟨*proof*⟩

lemma *oLog-omega-less*: ⟦*0* < *x*; *x* < *ε*$_0$⟧ ⟹ *oLog ω 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*: ⋀*g*. ∀ *n*. *g n* ∈ *A* ⟹ *oLimit g* ∈ *A*
and *unbounded*: ⋀*x*. ∃ *y*∈*A*. *x* < *y*

lemma (**in** *normal-set*) *less-next*: *x* < (*LEAST z*. *z* ∈ *A* ∧ *x* < *z*)
⟨*proof*⟩

lemma (**in** *normal-set*) *mem-next*: (*LEAST z*. *z* ∈ *A* ∧ *x* < *z*) ∈ *A*
⟨*proof*⟩

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

lemma *oLimit-mem-INTER*:
$[\![\forall\, n.\ normal\text{-}set\ (A\ n);\ \forall\, n.\ A\ (Suc\ n) \subseteq A\ n;$
$\forall\, n.\ f\ n \in A\ n;\ mono\ f]\!]$
$\Longrightarrow oLimit\ f \in (\bigcap n.\ A\ n)$
⟨*proof*⟩

lemma *normal-set-INTER*:
$[\![\forall\, n.\ normal\text{-}set\ (A\ n);\ \forall\, n.\ A\ (Suc\ n) \subseteq A\ n]\!] \Longrightarrow normal\text{-}set\ (\bigcap n.\ A\ n)$
⟨*proof*⟩

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*) ⇒ (*ordinal* ⇒ *ordinal*) **where**
ordering A = *ordinal-rec* (*LEAST z*. *z* ∈ *A*) (λ*p x*. *LEAST z*. *z* ∈ *A* ∧ *x* < *z*)

lemma *ordering-0*:
ordering A 0 = (*LEAST z*. *z* ∈ *A*)
⟨*proof*⟩

lemma *ordering-oSuc*:
ordering A (*oSuc x*) = (*LEAST z*. *z* ∈ *A* ∧ *ordering A x* < *z*)
⟨*proof*⟩

lemma (**in** *normal-set*) *normal-ordering*: *normal* (*ordering A*)
⟨*proof*⟩

lemma (**in** *normal-set*) *ordering-oLimit*:
ordering A (*oLimit f*) = *oLimit* (λ*n*. *ordering A* (*f n*))
⟨*proof*⟩

lemma (**in** *normal*) *ordering-range*: *ordering* (*range F*) = *F*
⟨*proof*⟩

lemma (**in** *normal-set*) *ordering-mem*: *ordering A x* ∈ *A*
⟨*proof*⟩

lemma (**in** *normal-set*) *range-ordering-lemma*:
∀ *y*. *y* ∈ *A* ⟶ *y* < *ordering A x* ⟶ *y* ∈ *range* (*ordering A*)
⟨*proof*⟩

lemma (**in** *normal-set*) *range-ordering*: *range* (*ordering A*) = *A*
⟨*proof*⟩

lemma *ordering-INTER-0*:
$[\![\forall\, n.\ normal\text{-}set\ (A\ n);\ \forall\, n.\ A\ (Suc\ n) \subseteq A\ n]\!]$

$\Longrightarrow$ *ordering* $(\bigcap n.\ A\ n)\ 0 = oLimit\ (\lambda n.\ ordering\ (A\ n)\ 0)$
⟨*proof*⟩

9.3 Critical ordinals

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

lemma *critical-set-0*:
critical-set A 0 = *A*
⟨*proof*⟩

lemma *critical-set-oSuc-lemma*:
critical-set A (*oSuc n*) =
critical-set A n ∩ *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*) = $(\bigcap n.$ *critical-set A* (*f n*))
⟨*proof*⟩

lemma *critical-set-mono*:
$x \leq y \Longrightarrow$ *critical-set A y* $\subseteq$ *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 $\Longrightarrow$ *normal-set* (*critical-set A x*)
⟨*proof*⟩

lemma *critical-set-oSuc*:
normal-set A
$\Longrightarrow$ *critical-set A* (*oSuc x*) = *range* (*oDeriv* (*ordering* (*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 = $(\lambda x.$ *ordering* (*critical-set* (*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*:
$0 < F\ 0 \Longrightarrow normal\ (\lambda x.\ oVeblen\ F\ x\ 0)$
⟨*proof*⟩

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

lemma (**in** *normal*) *range-oVeblen-subset*:
$x \leq y \Longrightarrow range\ (oVeblen\ F\ y) \subseteq range\ (oVeblen\ F\ x)$
⟨*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*:
$0 < z \Longrightarrow range\ (oVeblen\ F\ z) = \{x.\ \forall\, y{<}z.\ oVeblen\ F\ y\ x = x\}$
⟨*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* ((+) *a*) = ((+) (*a* ∗ ω))
⟨*proof*⟩

lemma *oVeblen-1-plus*: *oVeblen* ((+) *1*) *x* = ((+) (ω ∗∗ *x*))
⟨*proof*⟩

end