Lattices and Orders in Isabelle/HOL

Markus Wenzel
TU München

October 8, 2017

Abstract

We consider abstract structures of orders and lattices. Many fundamental concepts of lattice theory are developed, including dual structures, properties of bounds versus algebraic laws, lattice operations versus set-theoretic ones etc. We also give example instantiations of lattices and orders, such as direct products and function spaces. Well-known properties are demonstrated, like the Knaster-Tarski Theorem for complete lattices.

This formal theory development may serve as an example of applying Isabelle/HOL to the domain of mathematical reasoning about "axiomatic" structures. Apart from the simply-typed classical set-theory of HOL, we employ Isabelle's system of axiomatic type classes for expressing structures and functors in a light-weight manner. Proofs are expressed in the Isar language for readable formal proof, while aiming at its "best-style" of representing formal reasoning.

Contents

1 Orders

theory *Orders* **imports** *Main* **begin**

1.1 Ordered structures

We define several classes of ordered structures over some type $'a$ with relation $\sqsubseteq :: 'a \Rightarrow 'a \Rightarrow bool$. For a *quasi-order* that relation is required to be reflexive and transitive, for a *partial order* it also has to be anti-symmetric, while for a *linear order* all elements are required to be related (in either direction).

class *leq* =
  fixes $leq :: 'a \Rightarrow 'a \Rightarrow bool$ (**infixl** $\sqsubseteq$ *50*)

class *quasi-order* = *leq* +
  assumes *leq-refl* [*intro?*]: $x \sqsubseteq x$
  assumes *leq-trans* [*trans*]: $x \sqsubseteq y \Longrightarrow y \sqsubseteq z \Longrightarrow x \sqsubseteq z$

class *partial-order* = *quasi-order* +
  assumes *leq-antisym* [*trans*]: $x \sqsubseteq y \Longrightarrow y \sqsubseteq x \Longrightarrow x = y$

class *linear-order* = *partial-order* +
  assumes *leq-linear*: $x \sqsubseteq y \vee y \sqsubseteq x$

lemma *linear-order-cases*:
  $((x::'a::linear\text{-}order) \sqsubseteq y \Longrightarrow C) \Longrightarrow (y \sqsubseteq x \Longrightarrow C) \Longrightarrow C$
  ⟨*proof*⟩

1.2 Duality

The *dual* of an ordered structure is an isomorphic copy of the underlying type, with the $\sqsubseteq$ relation defined as the inverse of the original one.

datatype $'a$ *dual* = *dual* $'a$

primrec $undual :: 'a\ dual \Rightarrow 'a$ **where**
  undual-dual: $undual\ (dual\ x) = x$

instantiation *dual* :: (*leq*) *leq*
begin

definition
  leq-dual-def: $x' \sqsubseteq y' \equiv undual\ y' \sqsubseteq undual\ x'$

instance ⟨*proof*⟩

end

lemma *undual-leq* [*iff?*]: $(undual\ x' \sqsubseteq undual\ y') = (y' \sqsubseteq x')$
  ⟨*proof*⟩

lemma *dual-leq* [*iff?*]: $(dual\ x \sqsubseteq dual\ y) = (y \sqsubseteq x)$
⟨*proof*⟩

Functions *dual* and *undual* are inverse to each other; this entails the following fundamental properties.

lemma *dual-undual* [*simp*]: $dual\ (undual\ x') = x'$
⟨*proof*⟩

lemma *undual-dual-id* [*simp*]: $undual\ o\ dual = id$
⟨*proof*⟩

lemma *dual-undual-id* [*simp*]: $dual\ o\ undual = id$
⟨*proof*⟩

Since *dual* (and *undual*) are both injective and surjective, the basic logical connectives (equality, quantification etc.) are transferred as follows.

lemma *undual-equality* [*iff?*]: $(undual\ x' = undual\ y') = (x' = y')$
⟨*proof*⟩

lemma *dual-equality* [*iff?*]: $(dual\ x = dual\ y) = (x = y)$
⟨*proof*⟩

lemma *dual-ball* [*iff?*]: $(\forall x \in A.\ P\ (dual\ x)) = (\forall x' \in dual\ `\ A.\ P\ x')$
⟨*proof*⟩

lemma *range-dual* [*simp*]: *surj dual*
⟨*proof*⟩

lemma *dual-all* [*iff?*]: $(\forall x.\ P\ (dual\ x)) = (\forall x'.\ P\ x')$
⟨*proof*⟩

lemma *dual-ex*: $(\exists x.\ P\ (dual\ x)) = (\exists x'.\ P\ x')$
⟨*proof*⟩

lemma *dual-Collect*: $\{dual\ x|\ x.\ P\ (dual\ x)\} = \{x'.\ P\ x'\}$
⟨*proof*⟩

1.3 Transforming orders

1.3.1 Duals

The classes of quasi, partial, and linear orders are all closed under formation of dual structures.

instance *dual* :: (*quasi-order*) *quasi-order*
⟨*proof*⟩

instance *dual* :: (*partial-order*) *partial-order*
⟨*proof*⟩

instance *dual* :: (*linear-order*) *linear-order*
⟨*proof*⟩

1.3.2 Binary products

The classes of quasi and partial orders are closed under binary products. Note that the direct product of linear orders need *not* be linear in general.

instantiation *prod* :: (*leq*, *leq*) *leq*
begin

definition
leq-prod-def: $p \sqsubseteq q \equiv fst\ p \sqsubseteq fst\ q \wedge snd\ p \sqsubseteq snd\ q$

instance ⟨*proof*⟩

end

lemma *leq-prodI* [*intro?*]:
$fst\ p \sqsubseteq fst\ q \Longrightarrow snd\ p \sqsubseteq snd\ q \Longrightarrow p \sqsubseteq q$
⟨*proof*⟩

lemma *leq-prodE* [*elim?*]:
$p \sqsubseteq q \Longrightarrow (fst\ p \sqsubseteq fst\ q \Longrightarrow snd\ p \sqsubseteq snd\ q \Longrightarrow C) \Longrightarrow C$
⟨*proof*⟩

instance *prod* :: (*quasi-order*, *quasi-order*) *quasi-order*
⟨*proof*⟩

instance *prod* :: (*partial-order*, *partial-order*) *partial-order*
⟨*proof*⟩

1.3.3 General products

The classes of quasi and partial orders are closed under general products (function spaces). Note that the direct product of linear orders need *not* be linear in general.

instantiation *fun* :: (*type*, *leq*) *leq*
begin

definition
leq-fun-def: $f \sqsubseteq g \equiv \forall x.\ f\ x \sqsubseteq g\ x$

instance ⟨*proof*⟩

end

lemma *leq-funI* [*intro?*]: $(\bigwedge x.\ f\ x \sqsubseteq g\ x) \Longrightarrow f \sqsubseteq g$
⟨*proof*⟩

lemma *leq-funD* [*dest?*]: $f \sqsubseteq g \Longrightarrow f\ x \sqsubseteq g\ x$
⟨*proof*⟩

instance *fun* :: (*type*, *quasi-order*) *quasi-order*
⟨*proof*⟩

instance *fun* :: (*type*, *partial-order*) *partial-order*
⟨*proof*⟩

end

2 Bounds

theory *Bounds* **imports** *Orders* **begin**

hide-const (**open**) *inf sup*

2.1 Infimum and supremum

Given a partial order, we define infimum (greatest lower bound) and supremum (least upper bound) wrt. $\sqsubseteq$ for two and for any number of elements.

definition
is-inf :: $'a$::*partial-order* $\Rightarrow 'a \Rightarrow 'a \Rightarrow$ *bool* **where**
is-inf $x\ y\ inf = (inf \sqsubseteq x \wedge inf \sqsubseteq y \wedge (\forall z.\ z \sqsubseteq x \wedge z \sqsubseteq y \longrightarrow z \sqsubseteq inf))$

definition
is-sup :: $'a$::*partial-order* $\Rightarrow 'a \Rightarrow 'a \Rightarrow$ *bool* **where**
is-sup $x\ y\ sup = (x \sqsubseteq sup \wedge y \sqsubseteq sup \wedge (\forall z.\ x \sqsubseteq z \wedge y \sqsubseteq z \longrightarrow sup \sqsubseteq z))$

definition
is-Inf :: $'a$::*partial-order set* $\Rightarrow 'a \Rightarrow$ *bool* **where**
is-Inf $A\ inf = ((\forall x \in A.\ inf \sqsubseteq x) \wedge (\forall z.\ (\forall x \in A.\ z \sqsubseteq x) \longrightarrow z \sqsubseteq inf))$

definition
is-Sup :: $'a$::*partial-order set* $\Rightarrow 'a \Rightarrow$ *bool* **where**
is-Sup $A\ sup = ((\forall x \in A.\ x \sqsubseteq sup) \wedge (\forall z.\ (\forall x \in A.\ x \sqsubseteq z) \longrightarrow sup \sqsubseteq z))$

These definitions entail the following basic properties of boundary elements.

lemma *is-infI* [*intro?*]: $inf \sqsubseteq x \Longrightarrow inf \sqsubseteq y \Longrightarrow$
$(\bigwedge z.\ z \sqsubseteq x \Longrightarrow z \sqsubseteq y \Longrightarrow z \sqsubseteq inf) \Longrightarrow$ *is-inf* $x\ y\ inf$
⟨*proof*⟩

lemma *is-inf-greatest* [*elim?*]:
is-inf $x\ y\ inf \Longrightarrow z \sqsubseteq x \Longrightarrow z \sqsubseteq y \Longrightarrow z \sqsubseteq inf$

⟨*proof*⟩

lemma *is-inf-lower* [*elim?*]:
$is\text{-}inf\ x\ y\ inf \Longrightarrow (inf \sqsubseteq x \Longrightarrow inf \sqsubseteq y \Longrightarrow C) \Longrightarrow C$
⟨*proof*⟩

lemma *is-supI* [*intro?*]: $x \sqsubseteq sup \Longrightarrow y \sqsubseteq sup \Longrightarrow$
$(\bigwedge z.\ x \sqsubseteq z \Longrightarrow y \sqsubseteq z \Longrightarrow sup \sqsubseteq z) \Longrightarrow is\text{-}sup\ x\ y\ sup$
⟨*proof*⟩

lemma *is-sup-least* [*elim?*]:
$is\text{-}sup\ x\ y\ sup \Longrightarrow x \sqsubseteq z \Longrightarrow y \sqsubseteq z \Longrightarrow sup \sqsubseteq z$
⟨*proof*⟩

lemma *is-sup-upper* [*elim?*]:
$is\text{-}sup\ x\ y\ sup \Longrightarrow (x \sqsubseteq sup \Longrightarrow y \sqsubseteq sup \Longrightarrow C) \Longrightarrow C$
⟨*proof*⟩

lemma *is-InfI* [*intro?*]: $(\bigwedge x.\ x \in A \Longrightarrow inf \sqsubseteq x) \Longrightarrow$
$(\bigwedge z.\ (\forall x \in A.\ z \sqsubseteq x) \Longrightarrow z \sqsubseteq inf) \Longrightarrow is\text{-}Inf\ A\ inf$
⟨*proof*⟩

lemma *is-Inf-greatest* [*elim?*]:
$is\text{-}Inf\ A\ inf \Longrightarrow (\bigwedge x.\ x \in A \Longrightarrow z \sqsubseteq x) \Longrightarrow z \sqsubseteq inf$
⟨*proof*⟩

lemma *is-Inf-lower* [*dest?*]:
$is\text{-}Inf\ A\ inf \Longrightarrow x \in A \Longrightarrow inf \sqsubseteq x$
⟨*proof*⟩

lemma *is-SupI* [*intro?*]: $(\bigwedge x.\ x \in A \Longrightarrow x \sqsubseteq sup) \Longrightarrow$
$(\bigwedge z.\ (\forall x \in A.\ x \sqsubseteq z) \Longrightarrow sup \sqsubseteq z) \Longrightarrow is\text{-}Sup\ A\ sup$
⟨*proof*⟩

lemma *is-Sup-least* [*elim?*]:
$is\text{-}Sup\ A\ sup \Longrightarrow (\bigwedge x.\ x \in A \Longrightarrow x \sqsubseteq z) \Longrightarrow sup \sqsubseteq z$
⟨*proof*⟩

lemma *is-Sup-upper* [*dest?*]:
$is\text{-}Sup\ A\ sup \Longrightarrow x \in A \Longrightarrow x \sqsubseteq sup$
⟨*proof*⟩

2.2 Duality

Infimum and supremum are dual to each other.

theorem *dual-inf* [*iff?*]:

is-inf (*dual* x) (*dual* y) (*dual sup*) = *is-sup* x y *sup*
⟨*proof*⟩

theorem *dual-sup* [*iff?*]:
is-sup (*dual* x) (*dual* y) (*dual inf*) = *is-inf* x y *inf*
⟨*proof*⟩

theorem *dual-Inf* [*iff?*]:
is-Inf (*dual* ' A) (*dual sup*) = *is-Sup* A *sup*
⟨*proof*⟩

theorem *dual-Sup* [*iff?*]:
is-Sup (*dual* ' A) (*dual inf*) = *is-Inf* A *inf*
⟨*proof*⟩

2.3 Uniqueness

Infima and suprema on partial orders are unique; this is mainly due to anti-symmetry of the underlying relation.

theorem *is-inf-uniq*: *is-inf* x y *inf* $\Longrightarrow$ *is-inf* x y *inf'* $\Longrightarrow$ *inf* = *inf'*
⟨*proof*⟩

theorem *is-sup-uniq*: *is-sup* x y *sup* $\Longrightarrow$ *is-sup* x y *sup'* $\Longrightarrow$ *sup* = *sup'*
⟨*proof*⟩

theorem *is-Inf-uniq*: *is-Inf* A *inf* $\Longrightarrow$ *is-Inf* A *inf'* $\Longrightarrow$ *inf* = *inf'*
⟨*proof*⟩

theorem *is-Sup-uniq*: *is-Sup* A *sup* $\Longrightarrow$ *is-Sup* A *sup'* $\Longrightarrow$ *sup* = *sup'*
⟨*proof*⟩

2.4 Related elements

The binary bound of related elements is either one of the argument.

theorem *is-inf-related* [*elim?*]: $x \sqsubseteq y \Longrightarrow$ *is-inf* x y x
⟨*proof*⟩

theorem *is-sup-related* [*elim?*]: $x \sqsubseteq y \Longrightarrow$ *is-sup* x y y
⟨*proof*⟩

2.5 General versus binary bounds

General bounds of two-element sets coincide with binary bounds.

theorem *is-Inf-binary*: *is-Inf* $\{x, y\}$ *inf* = *is-inf* x y *inf*
⟨*proof*⟩

theorem *is-Sup-binary*: *is-Sup* $\{x, y\}$ *sup* = *is-sup* x y *sup*

⟨*proof*⟩

2.6 Connecting general bounds

Either kind of general bounds is sufficient to express the other. The least upper bound (supremum) is the same as the the greatest lower bound of the set of all upper bounds; the dual statements holds as well; the dual statement holds as well.

theorem *Inf-Sup*: $is\text{-}Inf\ \{b.\ \forall\, a \in A.\ a \sqsubseteq b\}\ sup \Longrightarrow is\text{-}Sup\ A\ sup$
⟨*proof*⟩

theorem *Sup-Inf*: $is\text{-}Sup\ \{b.\ \forall\, a \in A.\ b \sqsubseteq a\}\ inf \Longrightarrow is\text{-}Inf\ A\ inf$
⟨*proof*⟩

end

3 Lattices

theory *Lattice* **imports** *Bounds* **begin**

3.1 Lattice operations

A *lattice* is a partial order with infimum and supremum of any two elements (thus any *finite* number of elements have bounds as well).

class *lattice* =
  assumes *ex-inf*: $\exists\, inf.\ is\text{-}inf\ x\ y\ inf$
  assumes *ex-sup*: $\exists\, sup.\ is\text{-}sup\ x\ y\ sup$

The $\sqcap$ (meet) and $\sqcup$ (join) operations select such infimum and supremum elements.

definition
  meet :: $'a{::}lattice \Rightarrow\ 'a \Rightarrow\ 'a$ (**infixl** $\sqcap$ *70*) **where**
  $x \sqcap y = (THE\ inf.\ is\text{-}inf\ x\ y\ inf)$
definition
  join :: $'a{::}lattice \Rightarrow\ 'a \Rightarrow\ 'a$ (**infixl** $\sqcup$ *65*) **where**
  $x \sqcup y = (THE\ sup.\ is\text{-}sup\ x\ y\ sup)$

Due to unique existence of bounds, the lattice operations may be exhibited as follows.

lemma *meet-equality* [*elim?*]: $is\text{-}inf\ x\ y\ inf \Longrightarrow x \sqcap y = inf$
⟨*proof*⟩

lemma *meetI* [*intro?*]:
  $inf \sqsubseteq x \Longrightarrow inf \sqsubseteq y \Longrightarrow (\bigwedge z.\ z \sqsubseteq x \Longrightarrow z \sqsubseteq y \Longrightarrow z \sqsubseteq inf) \Longrightarrow x \sqcap y = inf$
  ⟨*proof*⟩

lemma *join-equality* [*elim?*]: *is-sup* x y *sup* $\Longrightarrow x \sqcup y = sup$
⟨*proof*⟩

lemma *joinI* [*intro?*]: $x \sqsubseteq sup \Longrightarrow y \sqsubseteq sup \Longrightarrow$
$(\bigwedge z.\ x \sqsubseteq z \Longrightarrow y \sqsubseteq z \Longrightarrow sup \sqsubseteq z) \Longrightarrow x \sqcup y = sup$
⟨*proof*⟩

The $\sqcap$ and $\sqcup$ operations indeed determine bounds on a lattice structure.

lemma *is-inf-meet* [*intro?*]: *is-inf* x y $(x \sqcap y)$
⟨*proof*⟩

lemma *meet-greatest* [*intro?*]: $z \sqsubseteq x \Longrightarrow z \sqsubseteq y \Longrightarrow z \sqsubseteq x \sqcap y$
⟨*proof*⟩

lemma *meet-lower1* [*intro?*]: $x \sqcap y \sqsubseteq x$
⟨*proof*⟩

lemma *meet-lower2* [*intro?*]: $x \sqcap y \sqsubseteq y$
⟨*proof*⟩

lemma *is-sup-join* [*intro?*]: *is-sup* x y $(x \sqcup y)$
⟨*proof*⟩

lemma *join-least* [*intro?*]: $x \sqsubseteq z \Longrightarrow y \sqsubseteq z \Longrightarrow x \sqcup y \sqsubseteq z$
⟨*proof*⟩

lemma *join-upper1* [*intro?*]: $x \sqsubseteq x \sqcup y$
⟨*proof*⟩

lemma *join-upper2* [*intro?*]: $y \sqsubseteq x \sqcup y$
⟨*proof*⟩

3.2 Duality

The class of lattices is closed under formation of dual structures. This means that for any theorem of lattice theory, the dualized statement holds as well; this important fact simplifies many proofs of lattice theory.

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

Apparently, the $\sqcap$ and $\sqcup$ operations are dual to each other.

theorem *dual-meet* [*intro?*]: *dual* $(x \sqcap y) =$ *dual* $x \sqcup$ *dual* y
⟨*proof*⟩

theorem *dual-join* [*intro?*]: *dual* $(x \sqcup y) =$ *dual* $x \sqcap$ *dual* y
⟨*proof*⟩

3.3 Algebraic properties

The $\sqcap$ and $\sqcup$ operations have the following characteristic algebraic properties: associative (A), commutative (C), and absorptive (AB).

theorem *meet-assoc*: $(x \sqcap y) \sqcap z = x \sqcap (y \sqcap z)$
⟨*proof*⟩

theorem *join-assoc*: $(x \sqcup y) \sqcup z = x \sqcup (y \sqcup z)$
⟨*proof*⟩

theorem *meet-commute*: $x \sqcap y = y \sqcap x$
⟨*proof*⟩

theorem *join-commute*: $x \sqcup y = y \sqcup x$
⟨*proof*⟩

theorem *meet-join-absorb*: $x \sqcap (x \sqcup y) = x$
⟨*proof*⟩

theorem *join-meet-absorb*: $x \sqcup (x \sqcap y) = x$
⟨*proof*⟩

Some further algebraic properties hold as well. The property idempotent (I) is a basic algebraic consequence of (AB).

theorem *meet-idem*: $x \sqcap x = x$
⟨*proof*⟩

theorem *join-idem*: $x \sqcup x = x$
⟨*proof*⟩

Meet and join are trivial for related elements.

theorem *meet-related* [*elim?*]: $x \sqsubseteq y \Longrightarrow x \sqcap y = x$
⟨*proof*⟩

theorem *join-related* [*elim?*]: $x \sqsubseteq y \Longrightarrow x \sqcup y = y$
⟨*proof*⟩

3.4 Order versus algebraic structure

The $\sqcap$ and $\sqcup$ operations are connected with the underlying $\sqsubseteq$ relation in a canonical manner.

theorem *meet-connection*: $(x \sqsubseteq y) = (x \sqcap y = x)$
⟨*proof*⟩

theorem *join-connection*: $(x \sqsubseteq y) = (x \sqcup y = y)$
⟨*proof*⟩

The most fundamental result of the meta-theory of lattices is as follows (we do not prove it here).

Given a structure with binary operations $\sqcap$ and $\sqcup$ such that (A), (C), and (AB) hold (cf. §3.3). This structure represents a lattice, if the relation $x \sqsubseteq y$ is defined as $x \sqcap y = x$ (alternatively as $x \sqcup y = y$). Furthermore, infimum and supremum with respect to this ordering coincide with the original $\sqcap$ and $\sqcup$ operations.

3.5 Example instances

3.5.1 Linear orders

Linear orders with *minimum* and *maximum* operations are a (degenerate) example of lattice structures.

definition
minimum :: *'a::linear-order* ⇒ *'a* ⇒ *'a* **where**
minimum x y = (*if x* ⊑ *y then x else y*)
definition
maximum :: *'a::linear-order* ⇒ *'a* ⇒ *'a* **where**
maximum x y = (*if x* ⊑ *y then y else x*)

lemma *is-inf-minimum*: *is-inf x y* (*minimum x y*)
⟨*proof*⟩

lemma *is-sup-maximum*: *is-sup x y* (*maximum x y*)
⟨*proof*⟩

instance *linear-order* ⊆ *lattice*
⟨*proof*⟩

The lattice operations on linear orders indeed coincide with *minimum* and *maximum*.

theorem *meet-mimimum*: $x \sqcap y$ = *minimum x y*
⟨*proof*⟩

theorem *meet-maximum*: $x \sqcup y$ = *maximum x y*
⟨*proof*⟩

3.5.2 Binary products

The class of lattices is closed under direct binary products (cf. §1.3.2).

lemma *is-inf-prod*: *is-inf p q* (*fst p* ⊓ *fst q*, *snd p* ⊓ *snd q*)
⟨*proof*⟩

lemma *is-sup-prod*: *is-sup p q* (*fst p* ⊔ *fst q*, *snd p* ⊔ *snd q*)
⟨*proof*⟩

instance *prod* :: (*lattice*, *lattice*) *lattice*
⟨*proof*⟩

The lattice operations on a binary product structure indeed coincide with the products of the original ones.

theorem *meet-prod*: $p \sqcap q = (fst\ p \sqcap fst\ q,\ snd\ p \sqcap snd\ q)$
⟨*proof*⟩

theorem *join-prod*: $p \sqcup q = (fst\ p \sqcup fst\ q,\ snd\ p \sqcup snd\ q)$
⟨*proof*⟩

3.5.3 General products

The class of lattices is closed under general products (function spaces) as well (cf. §1.3.3).

lemma *is-inf-fun*: $is\text{-}inf\ f\ g\ (\lambda x.\ f\ x \sqcap g\ x)$
⟨*proof*⟩

lemma *is-sup-fun*: $is\text{-}sup\ f\ g\ (\lambda x.\ f\ x \sqcup g\ x)$
⟨*proof*⟩

instance *fun* :: (*type*, *lattice*) *lattice*
⟨*proof*⟩

The lattice operations on a general product structure (function space) indeed emerge by point-wise lifting of the original ones.

theorem *meet-fun*: $f \sqcap g = (\lambda x.\ f\ x \sqcap g\ x)$
⟨*proof*⟩

theorem *join-fun*: $f \sqcup g = (\lambda x.\ f\ x \sqcup g\ x)$
⟨*proof*⟩

3.6 Monotonicity and semi-morphisms

The lattice operations are monotone in both argument positions. In fact, monotonicity of the second position is trivial due to commutativity.

theorem *meet-mono*: $x \sqsubseteq z \Longrightarrow y \sqsubseteq w \Longrightarrow x \sqcap y \sqsubseteq z \sqcap w$
⟨*proof*⟩

theorem *join-mono*: $x \sqsubseteq z \Longrightarrow y \sqsubseteq w \Longrightarrow x \sqcup y \sqsubseteq z \sqcup w$
⟨*proof*⟩

A semi-morphisms is a function f that preserves the lattice operations in the following manner: $f\ (x \sqcap y) \sqsubseteq f\ x \sqcap f\ y$ and $f\ x \sqcup f\ y \sqsubseteq f\ (x \sqcup y)$, respectively. Any of these properties is equivalent with monotonicity.

theorem *meet-semimorph*:
$(\bigwedge x\ y.\ f\ (x \sqcap y) \sqsubseteq f\ x \sqcap f\ y) \equiv (\bigwedge x\ y.\ x \sqsubseteq y \Longrightarrow f\ x \sqsubseteq f\ y)$
⟨*proof*⟩

lemma *join-semimorph*:
$(\bigwedge x\ y.\ f\ x \sqcup f\ y \sqsubseteq f\ (x \sqcup y)) \equiv (\bigwedge x\ y.\ x \sqsubseteq y \Longrightarrow f\ x \sqsubseteq f\ y)$
⟨*proof*⟩

end

4 Complete lattices

theory *CompleteLattice* **imports** *Lattice* **begin**

4.1 Complete lattice operations

A *complete lattice* is a partial order with general (infinitary) infimum of any set of elements. General supremum exists as well, as a consequence of the connection of infinitary bounds (see §2.6).

class *complete-lattice* =
assumes *ex-Inf*: $\exists\, inf.\ is\text{-}Inf\ A\ inf$

theorem *ex-Sup*: $\exists\, sup{::}'a{::}complete\text{-}lattice.\ is\text{-}Sup\ A\ sup$
⟨*proof*⟩

The general $\bigsqcap$ (meet) and $\bigsqcup$ (join) operations select such infimum and supremum elements.

definition
Meet :: $'a{::}complete\text{-}lattice\ set \Rightarrow\ 'a$ ($\bigsqcap$- [*90*] *90*) **where**
$\bigsqcap A = (THE\ inf.\ is\text{-}Inf\ A\ inf)$
definition
Join :: $'a{::}complete\text{-}lattice\ set \Rightarrow\ 'a$ ($\bigsqcup$- [*90*] *90*) **where**
$\bigsqcup A = (THE\ sup.\ is\text{-}Sup\ A\ sup)$

Due to unique existence of bounds, the complete lattice operations may be exhibited as follows.

lemma *Meet-equality* [*elim?*]: $is\text{-}Inf\ A\ inf \Longrightarrow \bigsqcap A = inf$
⟨*proof*⟩

lemma *MeetI* [*intro?*]:
$(\bigwedge a.\ a \in A \Longrightarrow inf \sqsubseteq a) \Longrightarrow$
$(\bigwedge b.\ \forall a \in A.\ b \sqsubseteq a \Longrightarrow b \sqsubseteq inf) \Longrightarrow$
$\bigsqcap A = inf$
⟨*proof*⟩

lemma *Join-equality* [*elim?*]: $is\text{-}Sup\ A\ sup \Longrightarrow \bigsqcup A = sup$
⟨*proof*⟩

lemma *JoinI* [*intro?*]:
$(\bigwedge a.\ a \in A \Longrightarrow a \sqsubseteq sup) \Longrightarrow$
$(\bigwedge b.\ \forall\, a \in A.\ a \sqsubseteq b \Longrightarrow sup \sqsubseteq b) \Longrightarrow$
$\bigsqcup A = sup$
$\langle proof \rangle$

The $\bigsqcap$ and $\bigsqcup$ operations indeed determine bounds on a complete lattice structure.

lemma *is-Inf-Meet* [*intro?*]: *is-Inf* A $(\bigsqcap A)$
$\langle proof \rangle$

lemma *Meet-greatest* [*intro?*]: $(\bigwedge a.\ a \in A \Longrightarrow x \sqsubseteq a) \Longrightarrow x \sqsubseteq \bigsqcap A$
$\langle proof \rangle$

lemma *Meet-lower* [*intro?*]: $a \in A \Longrightarrow \bigsqcap A \sqsubseteq a$
$\langle proof \rangle$

lemma *is-Sup-Join* [*intro?*]: *is-Sup* A $(\bigsqcup A)$
$\langle proof \rangle$

lemma *Join-least* [*intro?*]: $(\bigwedge a.\ a \in A \Longrightarrow a \sqsubseteq x) \Longrightarrow \bigsqcup A \sqsubseteq x$
$\langle proof \rangle$
lemma *Join-lower* [*intro?*]: $a \in A \Longrightarrow a \sqsubseteq \bigsqcup A$
$\langle proof \rangle$

4.2 The Knaster-Tarski Theorem

The Knaster-Tarski Theorem (in its simplest formulation) states that any monotone function on a complete lattice has a least fixed-point (see [**?**, pages 93–94] for example). This is a consequence of the basic boundary properties of the complete lattice operations.

theorem *Knaster-Tarski*:
assumes *mono*: $\bigwedge x\ y.\ x \sqsubseteq y \Longrightarrow f\ x \sqsubseteq f\ y$
obtains $a :: 'a::complete\text{-}lattice$ **where**
$f\ a = a$ **and** $\bigwedge a'.\ f\ a' = a' \Longrightarrow a \sqsubseteq a'$
$\langle proof \rangle$

theorem *Knaster-Tarski-dual*:
assumes *mono*: $\bigwedge x\ y.\ x \sqsubseteq y \Longrightarrow f\ x \sqsubseteq f\ y$
obtains $a :: 'a::complete\text{-}lattice$ **where**
$f\ a = a$ **and** $\bigwedge a'.\ f\ a' = a' \Longrightarrow a' \sqsubseteq a$
$\langle proof \rangle$

4.3 Bottom and top elements

With general bounds available, complete lattices also have least and greatest elements.

definition
bottom :: *'a::complete-lattice* (⊥) **where**
$\bot = \bigsqcap UNIV$

definition
top :: *'a::complete-lattice* (⊤) **where**
$\top = \bigsqcup UNIV$

lemma *bottom-least* [*intro?*]: $\bot \sqsubseteq x$
⟨*proof*⟩

lemma *bottomI* [*intro?*]: $(\bigwedge a.\ x \sqsubseteq a) \Longrightarrow \bot = x$
⟨*proof*⟩

lemma *top-greatest* [*intro?*]: $x \sqsubseteq \top$
⟨*proof*⟩

lemma *topI* [*intro?*]: $(\bigwedge a.\ a \sqsubseteq x) \Longrightarrow \top = x$
⟨*proof*⟩

4.4 Duality

The class of complete lattices is closed under formation of dual structures.

instance *dual* :: (*complete-lattice*) *complete-lattice*
⟨*proof*⟩

Apparently, the $\bigsqcap$ and $\bigsqcup$ operations are dual to each other.

theorem *dual-Meet* [*intro?*]: $dual\ (\bigsqcap A) = \bigsqcup (dual\ `\ A)$
⟨*proof*⟩

theorem *dual-Join* [*intro?*]: $dual\ (\bigsqcup A) = \bigsqcap (dual\ `\ A)$
⟨*proof*⟩

Likewise are ⊥ and ⊤ duals of each other.

theorem *dual-bottom* [*intro?*]: $dual\ \bot = \top$
⟨*proof*⟩

theorem *dual-top* [*intro?*]: $dual\ \top = \bot$
⟨*proof*⟩

4.5 Complete lattices are lattices

Complete lattices (with general bounds available) are indeed plain lattices as well. This holds due to the connection of general versus binary bounds that has been formally established in §2.5.

lemma *is-inf-binary*: *is-inf* x y ($\bigsqcap\{x,\ y\}$)
⟨*proof*⟩

lemma *is-sup-binary*: *is-sup* x y ($\bigsqcup\{x,\ y\}$)
⟨*proof*⟩

instance *complete-lattice* $\subseteq$ *lattice*
⟨*proof*⟩

theorem *meet-binary*: $x \sqcap y = \bigsqcap\{x,\ y\}$
⟨*proof*⟩

theorem *join-binary*: $x \sqcup y = \bigsqcup\{x,\ y\}$
⟨*proof*⟩

4.6 Complete lattices and set-theory operations

The complete lattice operations are (anti) monotone wrt. set inclusion.

theorem *Meet-subset-antimono*: $A \subseteq B \Longrightarrow \bigsqcap B \sqsubseteq \bigsqcap A$
⟨*proof*⟩

theorem *Join-subset-mono*: $A \subseteq B \Longrightarrow \bigsqcup A \sqsubseteq \bigsqcup B$
⟨*proof*⟩

Bounds over unions of sets may be obtained separately.

theorem *Meet-Un*: $\bigsqcap(A \cup B) = \bigsqcap A \sqcap \bigsqcap B$
⟨*proof*⟩

theorem *Join-Un*: $\bigsqcup(A \cup B) = \bigsqcup A \sqcup \bigsqcup B$
⟨*proof*⟩

Bounds over singleton sets are trivial.

theorem *Meet-singleton*: $\bigsqcap\{x\} = x$
⟨*proof*⟩

theorem *Join-singleton*: $\bigsqcup\{x\} = x$
⟨*proof*⟩

Bounds over the empty and universal set correspond to each other.

theorem *Meet-empty*: $\bigsqcap\{\} = \bigsqcup UNIV$
⟨*proof*⟩

theorem *Join-empty*: $\bigsqcup\{\} = \bigsqcap UNIV$
⟨*proof*⟩

end