Lattice Properties

Viorel Preoteasa

March 17, 2025

Abstract

This formalization introduces and collects some algebraic structures based on lattices and complete lattices for use in other developments. The structures introduced are modular, and lattice ordered groups. In addition to the results proved for the new lattices, this formalization also introduces theorems about latices and complete lattices in general.

Contents

1 Overview

Section 2 introduces well founded and transitive relations. Section 3 introduces some properties about fixpoints of monotonic application which maps monotonic functions to monotonic functions. The most important property is that such a monotonic application has the least fixpoint monotonic. Section 4 introduces conjunctive, disjunctive, universally conjunctive, and universally disjunctive functions. In section 5 some simplification lemmas for alttices are proved. Section 6 introduces modular lattices and proves some

properties about them and about distributive lattices. The main result of this section is that a lattice is distributive if and only if it satisfies

$$\forall x\ y\ z : x \sqcap z = y \sqcap z \wedge x \sqcup z = y \sqcup z \longrightarrow x = y$$

Section 7 introduces lattice ordered groups and some of their properties. The most important is that they are distributive lattices, and this property is proved using the results from Section 5.

2 Well founded and transitive relations

theory *WellFoundedTransitive*
imports *Main*
begin

class *transitive* = *ord* +
  assumes *order-trans1*: $x < y \Longrightarrow y < z \Longrightarrow x < z$
  and *less-eq-def*: $x \leq y \longleftrightarrow x = y \vee x < y$
begin

lemma *eq-less-eq* [*simp*]:
  $x = y \Longrightarrow x \leq y$
  ⟨*proof*⟩

lemma *order-trans2* [*simp*]:
  $x \leq y \Longrightarrow y < z \Longrightarrow x < z$
  ⟨*proof*⟩

lemma *order-trans3*:
  $x < y \Longrightarrow y \leq z \Longrightarrow x < z$
  ⟨*proof*⟩
end

class *well-founded* = *ord* +
  assumes *less-induct1* [*case-names less*]: $(!!x\ .\ (!!y\ .\ y < x \Longrightarrow P\ y) \Longrightarrow P\ x) \Longrightarrow P\ a$

class *well-founded-transitive* = *transitive* + *well-founded*

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

definition
  less-pair-def: $a < b \longleftrightarrow$ *fst a* $<$ *fst b* $\vee$ (*fst a* = *fst b* $\wedge$ *snd a* $<$ *snd b*)

definition
  less-eq-pair-def: $(a::('a::ord * 'b::ord)) <= b \longleftrightarrow a = b \vee a < b$
instance ⟨*proof*⟩
end

instantiation *prod*:: (*transitive*, *transitive*) *transitive*
begin
instance ⟨*proof*⟩
end

instantiation *prod*:: (*well-founded*, *well-founded*) *well-founded*
begin
instance ⟨*proof*⟩
end

instantiation *prod*:: (*well-founded-transitive*, *well-founded-transitive*) *well-founded-transitive*
begin
instance ⟨*proof*⟩
end

instantiation *nat* :: *transitive*
begin

instance ⟨*proof*⟩
end

instantiation *nat*:: *well-founded*
begin
instance ⟨*proof*⟩
end

instantiation *nat*:: *well-founded-transitive*
begin
instance ⟨*proof*⟩
end

end

3 Fixpoints and Complete Lattices

theory *Complete-Lattice-Prop*
imports *WellFoundedTransitive*
begin

This theory introduces some results about fixpoints of functions on complete lattices. The main result is that a monotonic function mapping momotonic functions to monotonic functions has the least fixpoint monotonic.

context *complete-lattice* **begin**

lemma *inf-Inf*: **assumes** *nonempty*: $A \neq \{\}$
shows *inf* x (*Inf* A) = *Inf* ((*inf* x) ‘ A)
⟨*proof*⟩

end

definition
mono-mono F = (*mono F* ∧ (∀ *f* . *mono f* ⟶ *mono* (*F f*)))

theorem *lfp-mono* [*simp*]:
mono-mono F ⟹ *mono* (*lfp F*)
⟨*proof*⟩

lemma *gfp-ordinal-induct*:
fixes *f* :: *'a*::*complete-lattice* => *'a*
assumes *mono*: *mono f*
and *P-f*: !!*S*. *P S* ==> *P* (*f S*)
and *P-Union*: !!*M*. ∀ *S*∈*M*. *P S* ==> *P* (*Inf M*)
shows *P* (*gfp f*)
⟨*proof*⟩
theorem *gfp-mono* [*simp*]:
mono-mono F ⟹ *mono* (*gfp F*)
⟨*proof*⟩

context *complete-lattice* **begin**

definition
Sup-less x (*w*::*'b*::*well-founded*) = *Sup* {*y* ::*'a* . ∃ *v* < *w* . *y* = *x v*}

lemma *Sup-less-upper*:
v < *w* ⟹ *P v* ≤ *Sup-less P w*
⟨*proof*⟩

lemma *Sup-less-least*:
(!! *v* . *v* < *w* ⟹ *P v* ≤ *Q*) ⟹ *Sup-less P w* ≤ *Q*
⟨*proof*⟩

end

lemma *Sup-less-fun-eq*:
((*Sup-less P w*) *i*) = (*Sup-less* (λ *v* . *P v i*)) *w*
⟨*proof*⟩

theorem *fp-wf-induction*:
f x = *x* ⟹ *mono f* ⟹ (∀ *w* . (*y w*) ≤ *f* (*Sup-less y w*)) ⟹ *Sup* (*range y*) ≤ *x*
⟨*proof*⟩

end

4 Conjunctive and Disjunctive Functions

theory *Conj-Disj*
imports *Main*
begin

This theory introduces the definitions and some properties for conjunctive, disjunctive, universally conjunctive, and universally disjunctive functions.

locale *conjunctive* =
  fixes *inf-b* :: $'b \Rightarrow 'b \Rightarrow 'b$
  and *inf-c* :: $'c \Rightarrow 'c \Rightarrow 'c$
  and *times-abc* :: $'a \Rightarrow 'b \Rightarrow 'c$
begin

definition
  conjunctive = $\{x\ .\ (\forall\ y\ z\ .\ \textit{times-abc}\ x\ (\textit{inf-b}\ y\ z) = \textit{inf-c}\ (\textit{times-abc}\ x\ y)\ (\textit{times-abc}\ x\ z))\}$

lemma *conjunctiveI*:
  assumes $(\bigwedge b\ c.\ \textit{times-abc}\ a\ (\textit{inf-b}\ b\ c) = \textit{inf-c}\ (\textit{times-abc}\ a\ b)\ (\textit{times-abc}\ a\ c))$
  shows $a \in \textit{conjunctive}$
  ⟨*proof*⟩

lemma *conjunctiveD*: $x \in \textit{conjunctive} \Longrightarrow \textit{times-abc}\ x\ (\textit{inf-b}\ y\ z) = \textit{inf-c}\ (\textit{times-abc}\ x\ y)\ (\textit{times-abc}\ x\ z)$
  ⟨*proof*⟩

end

interpretation *Apply*: *conjunctive* $\textit{inf}::'a::\textit{semilattice-inf} \Rightarrow 'a \Rightarrow 'a$
  $\textit{inf}::'b::\textit{semilattice-inf} \Rightarrow 'b \Rightarrow 'b\ \lambda\ f\ .\ f$
  ⟨*proof*⟩

interpretation *Comp*: *conjunctive* $\textit{inf}::('a::\textit{lattice} \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a)$
  $\textit{inf}::('a::\textit{lattice} \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a)\ (o)$
  ⟨*proof*⟩

lemma *Apply.conjunctive* = *Comp.conjunctive*
  ⟨*proof*⟩

locale *disjunctive* =
  fixes *sup-b* :: $'b \Rightarrow 'b \Rightarrow 'b$
  and *sup-c* :: $'c \Rightarrow 'c \Rightarrow 'c$
  and *times-abc* :: $'a \Rightarrow 'b \Rightarrow 'c$
begin

definition
disjunctive = {*x* . (∀ *y z* . *times-abc x* (*sup-b y z*) = *sup-c* (*times-abc x y*) (*times-abc x z*))}

lemma *disjunctiveI*:
assumes (⋀*b c*. *times-abc a* (*sup-b b c*) = *sup-c* (*times-abc a b*) (*times-abc a c*))
shows *a* ∈ *disjunctive*
⟨*proof*⟩

lemma *disjunctiveD*: *x* ∈ *disjunctive* ⟹ *times-abc x* (*sup-b y z*) = *sup-c* (*times-abc x y*) (*times-abc x z*)
⟨*proof*⟩

end

interpretation *Apply*: *disjunctive sup*::'*a*::*semilattice-sup* ⇒ '*a* ⇒ '*a*
sup::'*b*::*semilattice-sup* ⇒ '*b* ⇒ '*b* λ *f* . *f*
⟨*proof*⟩

interpretation *Comp*: *disjunctive sup*::('*a*::*lattice* ⇒ '*a*) ⇒ ('*a* ⇒ '*a*) ⇒ ('*a* ⇒ '*a*)
sup::('*a*::*lattice* ⇒ '*a*) ⇒ ('*a* ⇒ '*a*) ⇒ ('*a* ⇒ '*a*) (*o*)
⟨*proof*⟩

lemma *apply-comp-disjunctive*: *Apply.disjunctive* = *Comp.disjunctive*
⟨*proof*⟩

locale *Conjunctive* =
fixes *Inf-b* :: '*b set* ⇒ '*b*
and *Inf-c* :: '*c set* ⇒ '*c*
and *times-abc* :: '*a* ⇒ '*b* ⇒ '*c*
begin

definition
Conjunctive = {*x* . (∀ *X* . *times-abc x* (*Inf-b X*) = *Inf-c* ((*times-abc x*) ‘ *X*))}

lemma *ConjunctiveI*:
assumes ⋀*A*. *times-abc a* (*Inf-b A*) = *Inf-c* ((*times-abc a*) ‘ *A*)
shows *a* ∈ *Conjunctive*
⟨*proof*⟩

lemma *ConjunctiveD*:
assumes *a* ∈ *Conjunctive*
shows *times-abc a* (*Inf-b A*) = *Inf-c* ((*times-abc a*) ‘ *A*)
⟨*proof*⟩

end

interpretation *Apply*: *Conjunctive Inf Inf* $\lambda\ f\ .\ f$
⟨*proof*⟩

interpretation *Comp*: *Conjunctive Inf*::(('*a*::*complete-lattice* ⇒ '*a*) *set*) ⇒ ('*a* ⇒ '*a*)
Inf::(('*a*::*complete-lattice* ⇒ '*a*) *set*) ⇒ ('*a* ⇒ '*a*) (*o*)
⟨*proof*⟩

lemma *Apply.Conjunctive* = *Comp.Conjunctive*
⟨*proof*⟩

locale *Disjunctive* =
fixes *Sup-b* :: '*b set* ⇒ '*b*
and *Sup-c* :: '*c set* ⇒ '*c*
and *times-abc* :: '*a* ⇒ '*b* ⇒ '*c*
begin

definition
Disjunctive = {*x* . (∀ *X* . *times-abc x* (*Sup-b X*) = *Sup-c* ((*times-abc x*) ‘ *X*))}

lemma *DisjunctiveI*:
assumes ⋀*A*. *times-abc a* (*Sup-b A*) = *Sup-c* ((*times-abc a*) ‘ *A*)
shows *a* ∈ *Disjunctive*
⟨*proof*⟩

lemma *DisjunctiveD*: *x* ∈ *Disjunctive* ⟹ *times-abc x* (*Sup-b X*) = *Sup-c* ((*times-abc x*) ‘ *X*)
⟨*proof*⟩

end

interpretation *Apply*: *Disjunctive Sup Sup* $\lambda\ f\ .\ f$
⟨*proof*⟩

interpretation *Comp*: *Disjunctive Sup*::(('*a*::*complete-lattice* ⇒ '*a*) *set*) ⇒ ('*a* ⇒ '*a*)
Sup::(('*a*::*complete-lattice* ⇒ '*a*) *set*) ⇒ ('*a* ⇒ '*a*) (*o*)
⟨*proof*⟩

lemma *Apply.Disjunctive* = *Comp.Disjunctive*
⟨*proof*⟩

lemma [*simp*]: (*F*::'*a*::*complete-lattice* ⇒ '*b*::*complete-lattice*) ∈ *Apply.Conjunctive* ⟹ *F* ∈ *Apply.conjunctive*
⟨*proof*⟩

lemma [*simp*]: *F* ∈ *Apply.conjunctive* ⟹ *mono F*
⟨*proof*⟩

lemma [*simp*]: (*F*::'*a*::*complete-lattice* ⇒ '*b*::*complete-lattice*) ∈ *Apply.Conjunctive* ⟹ *F top* = *top*
⟨*proof*⟩

lemma [*simp*]: (*F*::'*a*::*complete-lattice* ⇒ '*b*::*complete-lattice*) ∈ *Apply.Disjunctive* ⟹ *F* ∈ *Apply.disjunctive*
⟨*proof*⟩

lemma [*simp*]: *F* ∈ *Apply.disjunctive* ⟹ *mono F*
⟨*proof*⟩

lemma [*simp*]: (*F*::'*a*::*complete-lattice* ⇒ '*b*::*complete-lattice*) ∈ *Apply.Disjunctive* ⟹ *F bot* = *bot*
⟨*proof*⟩

lemma *weak-fusion*: *h* ∈ *Apply.Disjunctive* ⟹ *mono f* ⟹ *mono g* ⟹
h o f ≤ *g o h* ⟹ *h* (*lfp f*) ≤ *lfp g*
⟨*proof*⟩

lemma *inf-Disj*: (λ (*x*::'*a*::*complete-distrib-lattice*) . *inf x y*) ∈ *Apply.Disjunctive*
⟨*proof*⟩

end

5 Simplification Lemmas for Lattices

theory *Lattice-Prop*
imports *Main*
begin

This theory introduces some simplification lemmas for semilattices and lattices

notation
inf (**infixl** ‹⊓› *70*) **and**
sup (**infixl** ‹⊔› *65*)

context *semilattice-inf* **begin**
lemma [*simp*]: (*x* ⊓ *y*) ⊓ *z* ≤ *x*
⟨*proof*⟩

lemma [*simp*]: *x* ⊓ *y* ⊓ *z* ≤ *y*
⟨*proof*⟩

lemma [*simp*]: *x* ⊓ (*y* ⊓ *z*) ≤ *y*
⟨*proof*⟩

lemma [*simp*]: *x* ⊓ (*y* ⊓ *z*) ≤ *z*
⟨*proof*⟩
end

context *semilattice-sup* **begin**

lemma [*simp*]: $x \leq x \sqcup y \sqcup z$
⟨*proof*⟩

lemma [*simp*]: $y \leq x \sqcup y \sqcup z$
⟨*proof*⟩

lemma [*simp*]: $y \leq x \sqcup (y \sqcup z)$
⟨*proof*⟩

lemma [*simp*]: $z \leq x \sqcup (y \sqcup z)$
⟨*proof*⟩
end

context *lattice* **begin**

lemma [*simp*]: $x \sqcap y \leq x \sqcup z$
⟨*proof*⟩

lemma [*simp*]: $y \sqcap x \leq x \sqcup z$
⟨*proof*⟩

lemma [*simp*]: $x \sqcap y \leq z \sqcup x$
⟨*proof*⟩

lemma [*simp*]: $y \sqcap x \leq z \sqcup x$
⟨*proof*⟩

end

end

6 Modular and Distributive Lattices

theory *Modular-Distrib-Lattice*
imports *Lattice-Prop*
begin

The main result of this theory is the fact that a lattice is distributive if and only if it satisfies the following property:

term $(\forall\ x\ y\ z\ .\ x \sqcap z = y \sqcap z \wedge x \sqcup z = y \sqcup z \Longrightarrow x = y)$

This result was proved by Bergmann in [1]. The formalization presented here is based on [3, 4].

class *modular-lattice* = *lattice* +
assumes *modular*: $x \leq y \Longrightarrow\ x \sqcup (y \sqcap z) = y \sqcap (x \sqcup z)$

context *distrib-lattice* **begin**
subclass *modular-lattice*
⟨*proof*⟩
end

context *lattice* **begin**
definition
d-aux a b c = (*a* ⊓ *b*) ⊔ (*b* ⊓ *c*) ⊔ (*c* ⊓ *a*)

lemma *d-b-c-a*: *d-aux b c a* = *d-aux a b c*
⟨*proof*⟩

lemma *d-c-a-b*: *d-aux c a b* = *d-aux a b c*
⟨*proof*⟩

definition
e-aux a b c = (*a* ⊔ *b*) ⊓ (*b* ⊔ *c*) ⊓ (*c* ⊔ *a*)

lemma *e-b-c-a*: *e-aux b c a* = *e-aux a b c*
⟨*proof*⟩

lemma *e-c-a-b*: *e-aux c a b* = *e-aux a b c*
⟨*proof*⟩

definition
a-aux a b c = (*a* ⊓ (*e-aux a b c*)) ⊔ (*d-aux a b c*)

definition
b-aux a b c = (*b* ⊓ (*e-aux a b c*)) ⊔ (*d-aux a b c*)

definition
c-aux a b c = (*c* ⊓ (*e-aux a b c*)) ⊔ (*d-aux a b c*)

lemma *b-a*: *b-aux a b c* = *a-aux b c a*
⟨*proof*⟩

lemma *c-a*: *c-aux a b c* = *a-aux c a b*
⟨*proof*⟩

lemma [*simp*]: *a-aux a b c* ≤ *e-aux a b c*
⟨*proof*⟩

lemma [*simp*]: *b-aux a b c* ≤ *e-aux a b c*
⟨*proof*⟩

lemma [*simp*]: *c-aux a b c* ≤ *e-aux a b c*
⟨*proof*⟩

lemma [*simp*]: *d-aux a b c* $\leq$ *a-aux a b c*
⟨*proof*⟩

lemma [*simp*]: *d-aux a b c* $\leq$ *b-aux a b c*
⟨*proof*⟩

lemma [*simp*]: *d-aux a b c* $\leq$ *c-aux a b c*
⟨*proof*⟩

lemma *a-meet-e*: $a \sqcap$ (*e-aux a b c*) $= a \sqcap (b \sqcup c)$
⟨*proof*⟩

lemma *b-meet-e*: $b \sqcap$ (*e-aux a b c*) $= b \sqcap (c \sqcup a)$
⟨*proof*⟩

lemma *c-meet-e*: $c \sqcap$ (*e-aux a b c*) $= c \sqcap (a \sqcup b)$
⟨*proof*⟩

lemma *a-join-d*: $a \sqcup$ *d-aux a b c* $= a \sqcup (b \sqcap c)$
⟨*proof*⟩

lemma *b-join-d*: $b \sqcup$ *d-aux a b c* $= b \sqcup (c \sqcap a)$
⟨*proof*⟩

end

context *lattice* **begin**
definition
no-distrib a b c $= (a \sqcap b \sqcup c \sqcap a < a \sqcap (b \sqcup c))$

definition
incomp x y $= (\neg\ x \leq y \wedge \neg\ y \leq x)$

definition
N5-lattice a b c $= (a \sqcap c = b \sqcap c \wedge a < b \wedge a \sqcup c = b \sqcup c)$

definition
M5-lattice a b c $= (a \sqcap b = b \sqcap c \wedge c \sqcap a = b \sqcap c \wedge a \sqcup b = b \sqcup c \wedge c \sqcup a = b \sqcup c \wedge a \sqcap b < a \sqcup b)$

lemma *M5-lattice-incomp*: *M5-lattice a b c* $\Longrightarrow$ *incomp a b*
⟨*proof*⟩
end

context *modular-lattice* **begin**

lemma *a-meet-d*: $a \sqcap$ (*d-aux a b c*) $= (a \sqcap b) \sqcup (c \sqcap a)$
⟨*proof*⟩

lemma *b-meet-d*: $b \sqcap (d\text{-}aux\ a\ b\ c) = (b \sqcap c) \sqcup (a \sqcap b)$
⟨*proof*⟩

lemma *c-meet-d*: $c \sqcap (d\text{-}aux\ a\ b\ c) = (c \sqcap a) \sqcup (b \sqcap c)$
⟨*proof*⟩

lemma *d-less-e*: $no\text{-}distrib\ a\ b\ c \Longrightarrow d\text{-}aux\ a\ b\ c < e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *a-meet-b-eq-d*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow a\text{-}aux\ a\ b\ c \sqcap b\text{-}aux\ a\ b\ c = d\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *b-meet-c-eq-d*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow b\text{-}aux\ a\ b\ c \sqcap c\text{-}aux\ a\ b\ c = d\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *c-meet-a-eq-d*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow c\text{-}aux\ a\ b\ c \sqcap a\text{-}aux\ a\ b\ c = d\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *a-def-equiv*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow a\text{-}aux\ a\ b\ c = (a \sqcup d\text{-}aux\ a\ b\ c) \sqcap e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *b-def-equiv*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow b\text{-}aux\ a\ b\ c = (b \sqcup d\text{-}aux\ a\ b\ c) \sqcap e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *c-def-equiv*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow c\text{-}aux\ a\ b\ c = (c \sqcup d\text{-}aux\ a\ b\ c) \sqcap e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *a-join-b-eq-e*: $d\text{-}aux\ a\ b\ c \leq e\text{-}aux\ a\ b\ c \Longrightarrow a\text{-}aux\ a\ b\ c \sqcup b\text{-}aux\ a\ b\ c = e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *b-join-c-eq-e*: $d\text{-}aux\ a\ b\ c <= e\text{-}aux\ a\ b\ c \Longrightarrow b\text{-}aux\ a\ b\ c \sqcup c\text{-}aux\ a\ b\ c = e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma *c-join-a-eq-e*: $d\text{-}aux\ a\ b\ c <= e\text{-}aux\ a\ b\ c \Longrightarrow c\text{-}aux\ a\ b\ c \sqcup a\text{-}aux\ a\ b\ c = e\text{-}aux\ a\ b\ c$
⟨*proof*⟩

lemma $no\text{-}distrib\ a\ b\ c \Longrightarrow incomp\ a\ b$
⟨*proof*⟩

lemma *M5-modular*: *no-distrib a b c* $\Longrightarrow$ *M5-lattice* (*a-aux a b c*) (*b-aux a b c*) (*c-aux a b c*)
⟨*proof*⟩

lemma *M5-modular-def*: *M5-lattice a b c* = ($a \sqcap b = b \sqcap c \wedge c \sqcap a = b \sqcap c \wedge a \sqcup b = b \sqcup c \wedge c \sqcup a = b \sqcup c \wedge a \sqcap b < a \sqcup b$)
⟨*proof*⟩

end

context *lattice* **begin**

lemma *not-modular-N5*: ($\neg$ *class.modular-lattice inf* ($(\leq)::'a \Rightarrow 'a \Rightarrow bool$) ($<$) *sup*) =
($\exists$ *a b c*::'*a* . *N5-lattice a b c*)
⟨*proof*⟩

lemma *not-distrib-N5-M5*: ($\neg$ *class.distrib-lattice* ($\sqcap$) ($(\leq)::'a \Rightarrow 'a \Rightarrow bool$) ($<$) ($\sqcup$)) =
(($\exists$ *a b c*::'*a* . *N5-lattice a b c*) $\vee$ ($\exists$ *a b c*::'*a* . *M5-lattice a b c*))
⟨*proof*⟩

lemma *distrib-not-N5-M5*: (*class.distrib-lattice* ($\sqcap$) ($(\leq)::'a \Rightarrow 'a \Rightarrow bool$) ($<$) ($\sqcup$)) =
(($\forall$ *a b c*::'*a* . $\neg$ *N5-lattice a b c*) $\wedge$ ($\forall$ *a b c*::'*a* . $\neg$ *M5-lattice a b c*))
⟨*proof*⟩

lemma *distrib-inf-sup-eq*:
(*class.distrib-lattice* ($\sqcap$) ($(\leq)::'a \Rightarrow 'a \Rightarrow bool$) ($<$) ($\sqcup$)) =
($\forall$ *x y z*::'*a* . $x \sqcap z = y \sqcap z \wedge x \sqcup z = y \sqcup z \longrightarrow x = y$)
⟨*proof*⟩
end

class *inf-sup-eq-lattice* = *lattice* +
assumes *inf-sup-eq*: $x \sqcap z = y \sqcap z \Longrightarrow x \sqcup z = y \sqcup z \Longrightarrow x = y$
begin
subclass *distrib-lattice*
⟨*proof*⟩
end

end

7 Lattice Orderd Groups

theory *Lattice-Ordered-Group*
imports *Modular-Distrib-Lattice*
begin

This theory introduces lattice ordered groups [2] and proves some results about them. The most important result is that a lattice ordered group is also a distributive lattice.

class *lgroup* = *group-add* + *lattice* +
assumes *add-order-preserving*: $a \leq b \Longrightarrow u + a + v \leq u + b + v$
begin

lemma *add-order-preserving-left*: $a \leq b \Longrightarrow u + a \leq u + b$
⟨*proof*⟩

lemma *add-order-preserving-right*: $a \leq b \Longrightarrow a + v \leq b + v$
⟨*proof*⟩

lemma *minus-order*: $-a \leq -b \Longrightarrow b \leq a$
⟨*proof*⟩

lemma *right-move-to-left*: $a + - c \leq b \Longrightarrow a \leq b + c$
⟨*proof*⟩

lemma *right-move-to-right*: $a \leq b + -c \Longrightarrow a + c \leq b$
⟨*proof*⟩

lemma [*simp*]: $(a \sqcap b) + c = (a + c) \sqcap (b + c)$
⟨*proof*⟩

lemma [*simp*]: $(a \sqcap b) - c = (a - c) \sqcap (b - c)$
⟨*proof*⟩

lemma *left-move-to-left*: $-c + a \leq b \Longrightarrow a \leq c + b$
⟨*proof*⟩

lemma *left-move-to-right*: $a \leq - c + b \Longrightarrow c + a \leq b$
⟨*proof*⟩

lemma [*simp*]: $c + (a \sqcap b) = (c + a) \sqcap (c + b)$
⟨*proof*⟩

lemma [*simp*]: $- (a \sqcap b) = (- a) \sqcup (- b)$
⟨*proof*⟩

lemma [*simp*]: $(a \sqcup b) + c = (a + c) \sqcup (b + c)$
⟨*proof*⟩

lemma [*simp*]: $c + (a \sqcup b) = (c + a) \sqcup (c + b)$
⟨*proof*⟩

lemma [*simp*]: $c - (a \sqcap b) = (c - a) \sqcup (c - b)$
⟨*proof*⟩

lemma [*simp*]: $(a \sqcup b) - c = (a - c) \sqcup (b - c)$
⟨*proof*⟩

lemma [*simp*]: $- (a \sqcup b) = (- a) \sqcap (- b)$
⟨*proof*⟩

lemma [*simp*]: $c - (a \sqcup b) = (c - a) \sqcap (c - b)$
⟨*proof*⟩

lemma *add-pos*: $0 \leq a \Longrightarrow b \leq b + a$
⟨*proof*⟩

lemma *add-pos-left*: $0 \leq a \Longrightarrow b \leq a + b$
⟨*proof*⟩

lemma *inf-sup*: $a - (a \sqcap b) + b = a \sqcup b$
⟨*proof*⟩

lemma *inf-sup-2*: $b = (a \sqcap b) - a + (a \sqcup b)$
⟨*proof*⟩

subclass *inf-sup-eq-lattice*
⟨*proof*⟩
end

end

References

[1] G. Bergmann. Zur axiomatik der elementargeometrie. *Monatshefte für Mathematik*, 36:269–284, 1929. 10.1007/BF02307616.

[2] G. Birkhoff. Lattice, ordered groups. *Ann. of Math. (2)*, 43:298–331, 1942.

[3] G. Birkhoff. *Lattice theory*. Third edition. American Mathematical Society Colloquium Publications, Vol. XXV. American Mathematical Society, Providence, R.I., 1967.

[4] S. Burris and H. P. Sankappanavar. *A course in universal algebra*, volume 78 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1981.