

# Topological semantics for paraconsistent and paracomplete logics

David Fuenmayor

March 3, 2021

## Abstract

We introduce a generalized topological semantics for paraconsistent and paracomplete logics by drawing upon early works on topological Boolean algebras (cf. works by Kuratowski, Zarycki, McKinsey & Tarski, etc.). In particular, this work exemplarily illustrates the shallow semantical embeddings approach (SSE) employing the proof assistant Isabelle/HOL. By means of the SSE technique we can effectively harness theorem provers, model finders and ‘hammers’ for reasoning with quantified non-classical logics.

## Contents

<b>1</b>	<b>Shallow embedding of a Boolean algebra of propositions</b>	<b>3</b>
1.1	Encoding Boolean operations . . . . .	4
1.2	Second-order operations and fixed-points . . . . .	4
1.3	Equality and atomicity . . . . .	5
1.4	Obtaining a complete Boolean Algebra . . . . .	6
1.5	Adding quantifiers (restricted and unrestricted) . . . . .	7
1.6	Relating quantifiers with further operators . . . . .	8
<b>2</b>	<b>Positive semantic conditions for operations</b>	<b>9</b>
2.1	Definitions (finitary case) . . . . .	10
2.2	Relations among conditions (finitary case) . . . . .	10
2.3	Definitions (infinitary case) . . . . .	11
2.4	Relations among conditions (infinitary case) . . . . .	12
2.5	Exploring the Barcan formula and its converse . . . . .	13
<b>3</b>	<b>Negative semantic conditions for operations</b>	<b>13</b>
3.1	Definitions and interrelations (finitary case) . . . . .	14
3.1.1	Principles of excluded middle, contradiction and explosion . . . . .	14
3.1.2	Contraposition rules . . . . .	14
3.1.3	Modus tollens rules . . . . .	15
3.1.4	Double negation introduction and elimination . . . . .	15
3.1.5	Normality and its dual . . . . .	16
3.1.6	De Morgan laws . . . . .	17
3.1.7	Contextual (strong) contraposition rule . . . . .	18
3.1.8	Local contraposition axioms . . . . .	19
3.1.9	Local modus tollens axioms . . . . .	20
3.1.10	Disjunctive syllogism . . . . .	20
3.2	Definitions and interrelations (infinitary case) . . . . .	21

<b>4</b>	<b>Topological operators</b>	<b>23</b>
4.1	Interior and closure . . . . .	23
4.1.1	Interior conditions . . . . .	23
4.1.2	Closure conditions . . . . .	24
4.1.3	Exploring dualities . . . . .	25
4.2	Frontier and border . . . . .	26
4.2.1	Frontier conditions . . . . .	26
4.2.2	Border conditions . . . . .	26
4.2.3	Relation with closure and interior . . . . .	28
4.2.4	Infinitary conditions . . . . .	31
4.3	Derivative and coherence . . . . .	31
4.3.1	Derivative conditions . . . . .	31
4.3.2	Infinitary conditions . . . . .	34
4.3.3	Coherence conditions . . . . .	34
<b>5</b>	<b>Generalized specialization orderings and Alexandrov topologies</b>	<b>36</b>
5.1	Specialization relations . . . . .	36
5.2	Alexandrov topology . . . . .	37
5.3	(Anti)symmetry and the separation axioms T0 and T1 . . . . .	38
<b>6</b>	<b>Frontier Algebra</b>	<b>39</b>
6.1	Basic properties . . . . .	39
6.2	Further properties . . . . .	41
<b>7</b>	<b>Frontier-based negation - Semantic conditions</b>	<b>43</b>
7.1	'Explosion' (ECQ), non-contradiction (LNC) and excluded middle (TND) . . . . .	43
7.2	Modus tollens (MT) . . . . .	44
7.3	Contraposition rules (CoP) . . . . .	45
7.4	Normality (negative) and its dual (nNor/nDNor) . . . . .	46
7.5	Double negation introduction/elimination (DNI/DNE) . . . . .	46
7.6	De Morgan laws . . . . .	47
7.7	Local contraposition axioms (lCoP) . . . . .	48
7.8	Disjunctive syllogism . . . . .	48
<b>8</b>	<b>Frontier-based negation - Fixed-points</b>	<b>49</b>
8.1	'Explosion' (ECQ) and excluded middle (TND) . . . . .	49
8.2	Contraposition rules . . . . .	49
8.3	Double negation introduction/elimination . . . . .	50
8.4	De Morgan laws . . . . .	51
8.5	Local contraposition axioms . . . . .	52
8.6	Disjunctive syllogism . . . . .	52
<b>9</b>	<b>Example application: Logics of Formal Inconsistency (LFIs)</b>	<b>53</b>
<b>10</b>	<b>Strict implication</b>	<b>56</b>
<b>11</b>	<b>Example application: Subintuitionistic and subminimal logics</b>	<b>59</b>
<b>12</b>	<b>Derivative algebra</b>	<b>63</b>
12.1	Basic properties . . . . .	64
12.2	Further properties . . . . .	65

<b>13 Example application: Logics of Formal Undeterminedness (LFUs)</b>	<b>68</b>
<b>14 Border algebra</b>	<b>70</b>
14.1 Basic properties . . . . .	70
<b>15 Closure algebra</b>	<b>71</b>
15.1 Basic properties . . . . .	71
<b>16 Interior algebra</b>	<b>72</b>
16.1 Basic properties . . . . .	72

```
theory sse-boolean-algebra
  imports Main
begin
```

```
declare[[syntax-ambiguity-warning=false]]
nitpick-params[assms=true, user-axioms=true, show-all, expect=genuine, format=3]
```

## 1 Shallow embedding of a Boolean algebra of propositions

In this section we present a shallow semantical embedding (SSE, cf. [1] and [2]) for a family of logics whose semantics is based upon extensions of (complete and atomic) Boolean algebras. The range of such logics is indeed very wide, including, as we will see, quantified paraconsistent and paracomplete (e.g. intuitionistic) logics. Aside from illustrating how the SSE approach works in practice we show how it allows us to effectively harness theorem provers, model finders and ‘hammers’ for reasoning with quantified non-classical logics. Proof reconstructions have deliberately been avoided. Most of the proofs (in fact, all one-liners) have been found using Sledgehammer.

Two notions play a fundamental role in this work: propositions and propositional functions. Propositions, qua sentence denotations, are modeled as objects of type  $w \Rightarrow bool$  (shortened as  $\sigma$ ). Propositional functions, as the name indicates, are basically anything with a (parametric) type  $t \Rightarrow \sigma$ .

We introduce a type  $w$  for the domain of points (aka. ‘worlds’, ‘states’, etc.).  $\sigma$  is a type alias for sets of points (i.e. propositions) encoded as characteristic functions.

```
typedecl w
type-synonym  $\sigma = w \Rightarrow bool$ 
```

In the sequel, we introduce the following naming convention for variables:

- (i) Latin letters (A, b, M, P, q, X, y, etc.) denote in general propositions (type  $\sigma$ ); however, we reserve letters D and S to denote sets of propositions (aka. domains/spaces) and the letters u, v and w to denote worlds/points.
- (ii) Greek letters (in particular  $\pi$ ) denote propositional functions (type  $t \Rightarrow \sigma$ ); among the latter we may employ the letters  $\varphi$ ,  $\psi$  and  $\eta$  to explicitly name those corresponding to unary connectives/operations (type  $\sigma \Rightarrow \sigma$ ).

## 1.1 Encoding Boolean operations

We start with an ordered algebra,

**abbreviation**  $sequ::\sigma\Rightarrow\sigma\Rightarrow bool$  (**infixr**  $\approx$  45) **where**  $A \approx B \equiv \forall w. (A w) \longleftrightarrow (B w)$

**abbreviation**  $subs::\sigma\Rightarrow\sigma\Rightarrow bool$  (**infixr**  $\preceq$  45) **where**  $A \preceq B \equiv \forall w. (A w) \longrightarrow (B w)$

**abbreviation**  $sup::\sigma\Rightarrow\sigma\Rightarrow bool$  (**infixr**  $\succeq$  45) **where**  $B \succeq A \equiv A \preceq B$

define meet and join by reusing HOL metalogical conjunction and disjunction,

**definition**  $meet::\sigma\Rightarrow\sigma\Rightarrow\sigma$  (**infixr**  $\wedge$  54) **where**  $A \wedge B \equiv \lambda w. (A w) \wedge (B w)$

**definition**  $join::\sigma\Rightarrow\sigma\Rightarrow\sigma$  (**infixr**  $\vee$  53) **where**  $A \vee B \equiv \lambda w. (A w) \vee (B w)$

and introduce further operations to obtain a Boolean 'algebra of propositions'.

**definition**  $top::\sigma$  ( $\top$ ) **where**  $\top \equiv \lambda w. True$

**definition**  $bottom::\sigma$  ( $\perp$ ) **where**  $\perp \equiv \lambda w. False$

**definition**  $impl::\sigma\Rightarrow\sigma\Rightarrow\sigma$  (**infixr**  $\rightarrow$  51) **where**  $A \rightarrow B \equiv \lambda w. (A w) \longrightarrow (B w)$

**definition**  $dimp::\sigma\Rightarrow\sigma\Rightarrow\sigma$  (**infixr**  $\leftrightarrow$  51) **where**  $A \leftrightarrow B \equiv \lambda w. (A w) \longleftrightarrow (B w)$

**definition**  $diff::\sigma\Rightarrow\sigma\Rightarrow\sigma$  (**infixr**  $\leftarrow$  51) **where**  $A \leftarrow B \equiv \lambda w. (A w) \wedge \neg(B w)$

**definition**  $compl::\sigma\Rightarrow\sigma$  ( $-$  [57]58) **where**  $-A \equiv \lambda w. \neg(A w)$

**named-theorems** *conn*

**declare**  $meet-def[conn]$   $join-def[conn]$   $top-def[conn]$   $bottom-def[conn]$

$impl-def[conn]$   $dimp-def[conn]$   $diff-def[conn]$   $compl-def[conn]$

Quite trivially, we can verify that the algebra satisfies some essential lattice properties.

**lemma**  $a \vee a \approx a$  **unfolding** *conn* **by** *simp*

**lemma**  $a \wedge a \approx a$  **unfolding** *conn* **by** *simp*

**lemma**  $a \preceq a \vee b$  **unfolding** *conn* **by** *simp*

**lemma**  $a \wedge b \preceq a$  **unfolding** *conn* **by** *simp*

**lemma**  $(a \wedge b) \vee b \approx b$  **unfolding** *conn* **by** *auto*

**lemma**  $a \wedge (a \vee b) \approx a$  **unfolding** *conn* **by** *auto*

**lemma**  $a \preceq c \implies b \preceq c \implies a \vee b \preceq c$  **unfolding** *conn* **by** *simp*

**lemma**  $c \preceq a \implies c \preceq b \implies c \preceq a \wedge b$  **unfolding** *conn* **by** *simp*

**lemma**  $a \preceq b \equiv (a \vee b) \approx b$  **unfolding** *conn* **by** *smt*

**lemma**  $b \preceq a \equiv (a \wedge b) \approx b$  **unfolding** *conn* **by** *smt*

**lemma**  $a \preceq c \implies b \preceq d \implies (a \vee b) \preceq (c \vee d)$  **unfolding** *conn* **by** *simp*

**lemma**  $a \preceq c \implies b \preceq d \implies (a \wedge b) \preceq (c \wedge d)$  **unfolding** *conn* **by** *simp*

## 1.2 Second-order operations and fixed-points

We define equality for propositional functions as follows.

**definition**  $equal-op::('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)\Rightarrow bool$  (**infix**  $\equiv$  60) **where**  $\varphi \equiv \psi \equiv \forall X. \varphi X \approx \psi X$

Moreover, we define some useful Boolean (2nd-order) operations on propositional functions,

**abbreviation**  $unionOp::('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)$  (**infixr**  $\sqcup$  61) **where**  $\varphi \sqcup \psi \equiv \lambda X. \varphi X \vee \psi X$

**abbreviation**  $interOp::('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)$  (**infixr**  $\sqcap$  62) **where**  $\varphi \sqcap \psi \equiv \lambda X. \varphi X \wedge \psi X$

**abbreviation**  $compOp::('t\Rightarrow\sigma)\Rightarrow('t\Rightarrow\sigma)$  ( $(-^c)$ ) **where**  $\varphi^c \equiv \lambda X. -\varphi X$

some of them explicitly targeting operations,

**definition**  $dual::(\sigma\Rightarrow\sigma)\Rightarrow(\sigma\Rightarrow\sigma)$  ( $(-^d)$ ) **where**  $\varphi^d \equiv \lambda X. -(\varphi(-X))$

and also define an useful operation (for technical purposes).

**definition**  $id::\sigma\Rightarrow\sigma$  (*id*) **where**  $id A \equiv A$

We now prove some useful lemmas (some of them may help the provers in their hard work).

**lemma** *comp-symm*:  $\varphi^c = \psi \implies \varphi = \psi^c$  **unfolding conn by blast**  
**lemma** *comp-invol*:  $\varphi^{cc} = \varphi$  **unfolding conn by blast**  
**lemma** *dual-symm*:  $(\varphi \equiv \psi^d) \implies (\psi \equiv \varphi^d)$  **unfolding dual-def conn by simp**  
**lemma** *dual-comp*:  $\varphi^{dc} = \varphi^{cd}$  **unfolding dual-def by simp**

**lemma** *id<sup>d</sup>*  $\equiv id$  **by** (*simp add: id-def dual-def equal-op-def conn*)  
**lemma** *id<sup>c</sup>*  $\equiv compl$  **by** (*simp add: id-def dual-def equal-op-def conn*)  
**lemma**  $(A \sqcup B)^d \equiv (A^d) \sqcap (B^d)$  **by** (*simp add: dual-def equal-op-def conn*)  
**lemma**  $(A \sqcup B)^c \equiv (A^c) \sqcap (B^c)$  **by** (*simp add: equal-op-def conn*)  
**lemma**  $(A \sqcap B)^d \equiv (A^d) \sqcup (B^d)$  **by** (*simp add: dual-def equal-op-def conn*)  
**lemma**  $(A \sqcap B)^c \equiv (A^c) \sqcup (B^c)$  **by** (*simp add: equal-op-def conn*)

The notion of a fixed point is a fundamental one. We speak of propositions being fixed points of operations. For a given operation we define in the usual way a fixed-point predicate for propositions.

**abbreviation** *fixedpoint*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow bool)$  (*fp*) **where** *fp*  $\varphi \equiv \lambda X. \varphi X \approx X$

**lemma** *fp-d*:  $(fp \ \varphi^d) X = (fp \ \varphi)(-X)$  **unfolding dual-def conn by auto**  
**lemma** *fp-c*:  $(fp \ \varphi^c) X = (X \approx -(\varphi X))$  **unfolding conn by auto**  
**lemma** *fp-dc*:  $(fp \ \varphi^{dc}) X = (X \approx \varphi(-X))$  **unfolding dual-def conn by auto**

Indeed, we can 'operationalize' this predicate by defining a fixed-point operator as follows:

**abbreviation** *fixedpoint-op*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $(-^{fp})$ ) **where**  $\varphi^{fp} \equiv \lambda X. (\varphi X) \leftrightarrow X$

**lemma** *ofp-c*:  $(\varphi^c)^{fp} \equiv (\varphi^{fp})^c$  **unfolding conn equal-op-def by auto**  
**lemma** *ofp-d*:  $(\varphi^d)^{fp} \equiv (\varphi^{fp})^{dc}$  **unfolding dual-def equal-op-def conn by auto**  
**lemma** *ofp-dc*:  $(\varphi^{dc})^{fp} \equiv (\varphi^{fp})^d$  **unfolding dual-def equal-op-def conn by auto**  
**lemma** *ofp-decomp*:  $\varphi^{fp} \equiv (id \sqcap \varphi) \sqcup ((id \sqcup \varphi)^c)$  **unfolding equal-op-def id-def conn by auto**  
**lemma** *ofp-invol*:  $(\varphi^{fp})^{fp} \equiv \varphi$  **unfolding conn equal-op-def by auto**

Fixed-point predicate and fixed-point operator are closely related.

**lemma** *fp-rel*:  $((fp \ \varphi) X) = (\varphi^{fp} X \approx \top)$  **unfolding conn by auto**  
**lemma** *fp-d-rel*:  $((fp \ \varphi^d) X) = (\varphi^{fp}(-X) \approx \top)$  **unfolding dual-def conn by auto**  
**lemma** *fp-c-rel*:  $((fp \ \varphi^c) X) = (\varphi^{fp} X \approx \perp)$  **unfolding conn by auto**  
**lemma** *fp-dc-rel*:  $((fp \ \varphi^{dc}) X) = (\varphi^{fp}(-X) \approx \perp)$  **unfolding dual-def conn by auto**

### 1.3 Equality and atomicity

We prove some facts about equality (which may help improve prover's performance).

**lemma** *eq-ext*:  $a \approx b \implies a = b$  **using ext by blast**  
**lemma** *eq-ext'*:  $a \equiv b \implies a = b$  **using ext unfolding equal-op-def by blast**  
**lemma** *meet-char*:  $a \preceq b \longleftrightarrow a \wedge b \approx a$  **unfolding conn by blast**  
**lemma** *join-char*:  $a \preceq b \longleftrightarrow a \vee b \approx b$  **unfolding conn by blast**

We can verify indeed that the algebra is atomic (in three different ways) by relying on the presence of primitive equality in HOL. A more general class of Boolean algebras could in principle be obtained in systems without primitive equality or by suitably restricting quantification over propositions (e.g. defining a topology and restricting quantifiers to open/closed sets).

**definition** *atom*  $a \equiv \neg(a \approx \perp) \wedge (\forall p. a \preceq p \vee a \preceq -p)$   
**lemma** *atomic1*:  $\forall w. \exists q. q w \wedge (\forall p. p w \longrightarrow q \preceq p)$  **using the-sym-eq-trivial by (metis (full-types))**  
**lemma** *atomic2*:  $\forall w. \exists q. q w \wedge atom(q)$  **using the-sym-eq-trivial by (metis (full-types) atom-def compl-def bottom-def)**

```

lemma atomic3:  $\forall p. \neg(p \approx \perp) \longrightarrow (\exists q. \text{atom}(q) \wedge q \preceq p)$  proof -
{ fix p
  { assume  $\neg(p \approx \perp)$ 
    hence  $\exists v. p \ v$  unfolding conn by simp
    then obtain w where  $1:p \ w$  by (rule exE)
    let  $?q=\lambda v. v = w$ 
    have  $2: \text{atom } ?q$  unfolding atom-def unfolding conn by simp
    have  $\forall v. ?q \ v \longrightarrow p \ v$  using  $1$  by simp
    hence  $3: ?q \preceq p$  by simp
    from  $2 \ 3$  have  $\exists q. \text{atom}(q) \wedge q \preceq p$  by blast
  } hence  $\neg(p \approx \perp) \longrightarrow (\exists q. \text{atom}(q) \wedge q \preceq p)$  by (rule impI)
} thus ?thesis by (rule allI)
qed

```

end

theory *sse-boolean-algebra-quantification*

imports *sse-boolean-algebra*

begin

hide-const(open) *List.list.Nil* **no-notation** *List.list.Nil* ( $\square$ )

hide-const(open) *Relation.converse* **no-notation** *Relation.converse* ( $(-^{-1})$  [1000] 999)

nitpick-params[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 1.4 Obtaining a complete Boolean Algebra

Our aim is to obtain a complete Boolean algebra which we can use to interpret quantified formulas (in the spirit of Boolean-valued models for set theory).

We start by defining infinite meet (infimum) and infinite join (supremum) operations,

**definition** *infimum*::  $(\sigma \Rightarrow \text{bool}) \Rightarrow \sigma$  ( $\bigwedge$ -) **where**  $\bigwedge S \equiv \lambda w. \forall X. S \ X \longrightarrow X \ w$

**definition** *supremum*::  $(\sigma \Rightarrow \text{bool}) \Rightarrow \sigma$  ( $\bigvee$ -) **where**  $\bigvee S \equiv \lambda w. \exists X. S \ X \wedge X \ w$

and show that the corresponding lattice is complete.

**abbreviation** *upper-bound*  $U \ S \equiv \forall X. (S \ X) \longrightarrow X \preceq U$

**abbreviation** *lower-bound*  $L \ S \equiv \forall X. (S \ X) \longrightarrow L \preceq X$

**abbreviation** *is-supremum*  $U \ S \equiv \text{upper-bound } U \ S \wedge (\forall X. \text{upper-bound } X \ S \longrightarrow U \preceq X)$

**abbreviation** *is-infimum*  $L \ S \equiv \text{lower-bound } L \ S \wedge (\forall X. \text{lower-bound } X \ S \longrightarrow X \preceq L)$

**lemma** *sup-char*: *is-supremum*  $\bigvee S \ S$  **unfolding** *supremum-def* **by** *auto*

**lemma** *sup-ext*:  $\forall S. \exists X. \text{is-supremum } X \ S$  **by** (*metis supremum-def*)

**lemma** *inf-char*: *is-infimum*  $\bigwedge S \ S$  **unfolding** *infimum-def* **by** *auto*

**lemma** *inf-ext*:  $\forall S. \exists X. \text{is-infimum } X \ S$  **by** (*metis infimum-def*)

We can check that being closed under supremum/infimum entails being closed under join/meet.

**abbreviation** *meet-closed*  $S \equiv \forall X \ Y. (S \ X \wedge S \ Y) \longrightarrow S(X \wedge Y)$

**abbreviation** *join-closed*  $S \equiv \forall X \ Y. (S \ X \wedge S \ Y) \longrightarrow S(X \vee Y)$

**abbreviation** *nonEmpty*  $S \equiv \exists x. S \ x$

**abbreviation** *contains*  $S \ D \equiv \forall X. D \ X \longrightarrow S \ X$

**abbreviation** *infimum-closed*  $S \equiv \forall D. \text{nonEmpty } D \wedge \text{contains } S \ D \longrightarrow S(\bigwedge D)$

**abbreviation** *supremum-closed*  $S \equiv \forall D. \text{nonEmpty } D \wedge \text{contains } S \ D \longrightarrow S(\bigvee D)$

**lemma** *inf-meet-closed*:  $\forall S. \text{infimum-closed } S \longrightarrow \text{meet-closed } S$  **proof** -

{ fix S

{ assume *inf-closed*: *infimum-closed* S

hence *meet-closed* S **proof** -

```

{ fix X::σ and Y::σ
  let ?D=λZ. Z=X ∨ Z=Y
  { assume S X ∧ S Y
    hence contains S ?D by simp
    moreover have nonEmpty ?D by auto
    ultimately have S(∧ ?D) using inf-closed by simp
    hence S(λw. ∀ Z. (Z=X ∨ Z=Y) → Z w) unfolding infimum-def by simp
    moreover have (λw. ∀ Z. (Z=X ∨ Z=Y) → Z w) = (λw. X w ∧ Y w) by auto
    ultimately have S(λw. X w ∧ Y w) by simp
  } hence (S X ∧ S Y) → S(X ∧ Y) unfolding conn by (rule impI)
  } thus ?thesis by simp qed
} hence infimum-closed S → meet-closed S by simp
} thus ?thesis by (rule allI)
qed
lemma sup-join-closed: ∀ P. supremum-closed P → join-closed P proof -
{ fix S
  { assume sup-closed: supremum-closed S
    hence join-closed S proof -
      { fix X::σ and Y::σ
        let ?D=λZ. Z=X ∨ Z=Y
        { assume S X ∧ S Y
          hence contains S ?D by simp
          moreover have nonEmpty ?D by auto
          ultimately have S(∨ ?D) using sup-closed by simp
          hence S(λw. ∃ Z. (Z=X ∨ Z=Y) ∧ Z w) unfolding supremum-def by simp
          moreover have (λw. ∃ Z. (Z=X ∨ Z=Y) ∧ Z w) = (λw. X w ∨ Y w) by auto
          ultimately have S(λw. X w ∨ Y w) by simp
        } hence (S X ∧ S Y) → S(X ∨ Y) unfolding conn by (rule impI)
        } thus ?thesis by simp qed
      } hence supremum-closed S → join-closed S by simp
    } thus ?thesis by (rule allI)
  }
}
qed

```

## 1.5 Adding quantifiers (restricted and unrestricted)

We can harness HOL to define quantification over individuals of arbitrary type (using polymorphism). These (unrestricted) quantifiers take a propositional function and give a proposition.

**abbreviation**  $mforall::('t \Rightarrow \sigma) \Rightarrow \sigma$  ( $\forall$  - [55]56) **where**  $\forall \pi \equiv \lambda w. \forall X. (\pi X) w$

**abbreviation**  $mexists::('t \Rightarrow \sigma) \Rightarrow \sigma$  ( $\exists$  - [55]56) **where**  $\exists \pi \equiv \lambda w. \exists X. (\pi X) w$

To improve readability, we introduce for them an useful binder notation.

**abbreviation**  $mforallB$  (**binder** $\forall$  [55]56) **where**  $\forall X. \pi X \equiv \forall \pi$

**abbreviation**  $mexistsB$  (**binder** $\exists$  [55]56) **where**  $\exists X. \pi X \equiv \exists \pi$

Moreover, we define restricted quantifiers which take a 'functional domain' as additional parameter. The latter is a propositional function that maps each element 'e' to the proposition 'e exists'.

**abbreviation**  $mforall-restr::('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \Rightarrow \sigma$  ( $\forall^R(-)$ ) **where**  $\forall^R(\delta)\pi \equiv \lambda w. \forall X. (\delta X) w \rightarrow (\pi X) w$

**abbreviation**  $mexists-restr::('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \Rightarrow \sigma$  ( $\exists^R(-)$ ) **where**  $\exists^R(\delta)\pi \equiv \lambda w. \exists X. (\delta X) w \wedge (\pi X) w$

## 1.6 Relating quantifiers with further operators

The following 'type-lifting' function is useful for converting sets into 'rigid' propositional functions.

**abbreviation**  $lift\text{-}conv::('t \Rightarrow bool) \Rightarrow ('t \Rightarrow \sigma) \ (\!| \cdot \!|) \ \mathbf{where} \ (\!|S\!|) \equiv \lambda X. \lambda w. S \ X$

We introduce an useful operator: the range of a propositional function (resp. restricted over a domain),

**definition**  $pfunRange::('t \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow bool) \ (Ra(-)) \ \mathbf{where} \ Ra(\pi) \equiv \lambda Y. \exists x. (\pi \ x) = Y$

**definition**  $pfunRange\text{-}restr::('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow bool) \Rightarrow (\sigma \Rightarrow bool) \ (Ra[\cdot|D]) \ \mathbf{where} \ Ra[\pi|D] \equiv \lambda Y. \exists x. (D \ x) \wedge (\pi \ x) = Y$

and check that taking infinite joins/meets (suprema/infima) over the range of a propositional function can be equivalently codified by using quantifiers. This is a quite useful simplifying relationship.

**lemma**  $Ra\text{-}all: \bigwedge Ra(\pi) = \forall \pi \ \mathbf{by} \ (metis \ (full\text{-}types) \ infimum\text{-}def \ pfunRange\text{-}def)$

**lemma**  $Ra\text{-}ex: \bigvee Ra(\pi) = \exists \pi \ \mathbf{by} \ (metis \ (full\text{-}types) \ pfunRange\text{-}def \ supremum\text{-}def)$

**lemma**  $Ra\text{-}restr\text{-}all: \bigwedge Ra[\pi|D] = \forall^R (\!|D\!|) \pi \ \mathbf{by} \ (metis \ (full\text{-}types) \ pfunRange\text{-}restr\text{-}def \ infimum\text{-}def)$

**lemma**  $Ra\text{-}restr\text{-}ex: \bigvee Ra[\pi|D] = \exists^R (\!|D\!|) \pi \ \mathbf{by} \ (metis \ pfunRange\text{-}restr\text{-}def \ supremum\text{-}def)$

We further introduce the positive (negative) restriction of a propositional function wrt. a domain,

**abbreviation**  $pfunRestr\text{-}pos::('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \ (\![\cdot]^P\!) \ \mathbf{where} \ [\pi|\delta]^P \equiv \lambda X. \lambda w. (\delta \ X) \ w \longrightarrow (\pi \ X) \ w$

**abbreviation**  $pfunRestr\text{-}neg::('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \Rightarrow ('t \Rightarrow \sigma) \ (\![\cdot]^N\!) \ \mathbf{where} \ [\pi|\delta]^N \equiv \lambda X. \lambda w. (\delta \ X) \ w \wedge (\pi \ X) \ w$

and check that some additional simplifying relationships obtain.

**lemma**  $all\text{-}restr: \forall^R (\delta) \pi = \forall [\pi|\delta]^P \ \mathbf{by} \ simp$

**lemma**  $ex\text{-}restr: \exists^R (\delta) \pi = \exists [\pi|\delta]^N \ \mathbf{by} \ simp$

**lemma**  $Ra\text{-}all\text{-}restr: \bigwedge Ra[\pi|D] = \forall [\pi|(\!|D\!|)]^P \ \mathbf{using} \ Ra\text{-}restr\text{-}all \ \mathbf{by} \ blast$

**lemma**  $Ra\text{-}ex\text{-}restr: \bigvee Ra[\pi|D] = \exists [\pi|(\!|D\!|)]^N \ \mathbf{by} \ (simp \ add: \ Ra\text{-}restr\text{-}ex)$

Observe that using these operators has the advantage of allowing for binder notation,

**lemma**  $\forall X. [\pi|\delta]^P \ X = \forall [\pi|\delta]^P \ \mathbf{by} \ simp$

**lemma**  $\exists X. [\pi|\delta]^N \ X = \exists [\pi|\delta]^N \ \mathbf{by} \ simp$

noting that extra care should be taken when working with complements or negations; always remember to switch P/N (positive/negative restriction) accordingly.

**lemma**  $\forall^R (\delta) \pi = \forall X. [\pi|\delta]^P \ X \ \mathbf{by} \ simp$

**lemma**  $\forall^R (\delta) \pi^c = \forall X. -[\pi|\delta]^N \ X \ \mathbf{by} \ (simp \ add: \ compl\text{-}def)$

**lemma**  $\exists^R (\delta) \pi = \exists X. [\pi|\delta]^N \ X \ \mathbf{by} \ simp$

**lemma**  $\exists^R (\delta) \pi^c = \exists X. -[\pi|\delta]^P \ X \ \mathbf{by} \ (simp \ add: \ compl\text{-}def)$

The previous definitions allow us to nicely characterize the interaction between function composition and (restricted) quantification:

**lemma**  $Ra\text{-}all\text{-}comp1: \forall (\pi \circ \gamma) = \forall [\pi|(\!|Ra \ \gamma\!|)]^P \ \mathbf{by} \ (metis \ comp\text{-}apply \ pfunRange\text{-}def)$

**lemma**  $Ra\text{-}all\text{-}comp2: \forall (\pi \circ \gamma) = \forall^R (\!|Ra \ \gamma\!|) \ \pi \ \mathbf{by} \ (metis \ comp\text{-}apply \ pfunRange\text{-}def)$

**lemma**  $Ra\text{-}ex\text{-}comp1: \exists (\pi \circ \gamma) = \exists [\pi|(\!|Ra \ \gamma\!|)]^N \ \mathbf{by} \ (metis \ comp\text{-}apply \ pfunRange\text{-}def)$

**lemma**  $Ra\text{-}ex\text{-}comp2: \exists (\pi \circ \gamma) = \exists^R (\!|Ra \ \gamma\!|) \ \pi \ \mathbf{by} \ (metis \ comp\text{-}apply \ pfunRange\text{-}def)$

This useful operator returns for a given domain of propositions the domain of their complements:



**definition** *dom-compl*:: $(\sigma \Rightarrow \text{bool}) \Rightarrow (\sigma \Rightarrow \text{bool})$   $((-^{-1}))$  **where**  $D^{-1} \equiv \lambda X. \exists Y. (D Y) \wedge (X = -Y)$   
**lemma** *dom-compl-def2*:  $D^{-1} = (\lambda X. D(-X))$  **unfolding** *dom-compl-def* **by** (*metis comp-symm fun-upd-same*)  
**lemma** *dom-compl-invol*:  $D = (D^{-1})^{-1}$  **unfolding** *dom-compl-def* **by** (*metis comp-symm fun-upd-same*)

We can now check an infinite variant of the De Morgan laws,

**lemma** *iDM-a*:  $-(\bigwedge S) = \bigvee S^{-1}$  **unfolding** *dom-compl-def2 infimum-def supremum-def* **using** *compl-def* **by force**

**lemma** *iDM-b*:  $-(\bigvee S) = \bigwedge S^{-1}$  **unfolding** *dom-compl-def2 infimum-def supremum-def* **using** *compl-def* **by force**

and some useful dualities regarding the range of propositional functions (restricted wrt. a domain).

**lemma** *Ra-compl*:  $Ra[\pi^c|D] = Ra[\pi|D]^{-1}$  **unfolding** *pfunRange-restr-def dom-compl-def* **by auto**

**lemma** *Ra-dual1*:  $Ra[\pi^d|D] = Ra[\pi|D^{-1}]^{-1}$  **unfolding** *pfunRange-restr-def dom-compl-def* **using** *dual-def* **by auto**

**lemma** *Ra-dual2*:  $Ra[\pi^d|D] = Ra[\pi^c|D^{-1}]$  **unfolding** *pfunRange-restr-def dom-compl-def* **using** *dual-def* **by auto**

**lemma** *Ra-dual3*:  $Ra[\pi^d|D]^{-1} = Ra[\pi|D^{-1}]$  **unfolding** *pfunRange-restr-def dom-compl-def* **using** *dual-def comp-symm* **by metis**

**lemma** *Ra-dual4*:  $Ra[\pi^d|D^{-1}] = Ra[\pi|D]^{-1}$  **using** *Ra-dual3 dual-symm* **by metis**

Finally, we check some facts concerning duality for quantifiers.

**lemma**  $\exists \pi^c = -(\forall \pi)$  **using** *compl-def* **by auto**

**lemma**  $\forall \pi^c = -(\exists \pi)$  **using** *compl-def* **by auto**

**lemma**  $\exists X. -\pi X = -(\forall X. \pi X)$  **using** *compl-def* **by auto**

**lemma**  $\forall X. -\pi X = -(\exists X. \pi X)$  **using** *compl-def* **by auto**

**lemma**  $\exists^R(\delta)\pi^c = -(\forall^R(\delta)\pi)$  **using** *compl-def* **by auto**

**lemma**  $\forall^R(\delta)\pi^c = -(\exists^R(\delta)\pi)$  **using** *compl-def* **by auto**

**lemma**  $\exists X. -[\pi|\delta]^P X = -(\forall X. [\pi|\delta]^P X)$  **using** *compl-def* **by auto**

**lemma**  $\forall X. -[\pi|\delta]^P X = -(\exists X. [\pi|\delta]^P X)$  **using** *compl-def* **by auto**

**lemma**  $\exists X. -[\pi|\delta]^N X = -(\forall X. [\pi|\delta]^N X)$  **using** *compl-def* **by auto**

**lemma**  $\forall X. -[\pi|\delta]^N X = -(\exists X. [\pi|\delta]^N X)$  **using** *compl-def* **by auto**

Warning: Do not switch P and N when passing to the dual form.

**lemma**  $\forall X. [\pi|\delta]^P X = -(\exists X. -[\pi|\delta]^N X)$  **nitpick oops** — wrong: counterexample

**lemma**  $\forall X. [\pi|\delta]^P X = -(\exists X. -[\pi|\delta]^P X)$  **using** *compl-def* **by auto** — correct

**end**

**theory** *sse-operation-positive*

**imports** *sse-boolean-algebra*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 2 Positive semantic conditions for operations

We define and interrelate some useful conditions on propositional functions which do not involve negative-like properties (hence 'positive'). We focus on propositional functions which correspond to unary connectives of the algebra (with type  $\sigma \Rightarrow \sigma$ ). We call such propositional functions 'operations'.

## 2.1 Definitions (finitary case)

Monotonicity (MONO).

**definition** *MONO*  $\varphi \equiv \forall A B. A \preceq B \longrightarrow \varphi A \preceq \varphi B$

**lemma** *MONO-ant*: *MONO*  $\varphi \implies \forall A B C. A \preceq B \longrightarrow \varphi(B \rightarrow C) \preceq \varphi(A \rightarrow C)$  **by** (*smt MONO-def conn*)

**lemma** *MONO-cons*: *MONO*  $\varphi \implies \forall A B C. A \preceq B \longrightarrow \varphi(C \rightarrow A) \preceq \varphi(C \rightarrow B)$  **by** (*smt MONO-def conn*)

**lemma** *MONO-dual*: *MONO*  $\varphi \implies \text{MONO } \varphi^d$  **by** (*smt MONO-def dual-def compl-def*)

Extensive/expansive (EXP) and its dual (dEXP), aka. 'contractive'.

**definition** *EXP*  $\varphi \equiv \forall A. A \preceq \varphi A$

**definition** *dEXP*  $\varphi \equiv \forall A. \varphi A \preceq A$

**lemma** *EXP-dual1*: *EXP*  $\varphi \implies \text{dEXP } \varphi^d$  **by** (*metis EXP-def dEXP-def dual-def compl-def*)

**lemma** *EXP-dual2*: *dEXP*  $\varphi \implies \text{EXP } \varphi^d$  **by** (*metis EXP-def dEXP-def dual-def compl-def*)

Idempotence (IDEM).

**definition** *IDEM*  $\varphi \equiv \forall A. (\varphi A) \approx \varphi(\varphi A)$

**definition** *IDEMa*  $\varphi \equiv \forall A. (\varphi A) \preceq \varphi(\varphi A)$

**definition** *IDEMb*  $\varphi \equiv \forall A. (\varphi A) \succeq \varphi(\varphi A)$

**lemma** *IDEM-dual1*: *IDEMa*  $\varphi \implies \text{IDEMb } \varphi^d$  **unfolding** *dual-def IDEMa-def IDEMb-def compl-def* **by** *auto*

**lemma** *IDEM-dual2*: *IDEMb*  $\varphi \implies \text{IDEMa } \varphi^d$  **unfolding** *dual-def IDEMa-def IDEMb-def compl-def* **by** *auto*

**lemma** *IDEM-dual*: *IDEM*  $\varphi = \text{IDEM } \varphi^d$  **by** (*metis IDEM-def IDEM-dual1 IDEM-dual2 IDEMa-def IDEMb-def dual-symm*)

Normality (NOR) and its dual (dNOR).

**definition** *NOR*  $\varphi \equiv (\varphi \perp) \approx \perp$

**definition** *dNOR*  $\varphi \equiv (\varphi \top) \approx \top$

**lemma** *NOR-dual1*: *NOR*  $\varphi = \text{dNOR } \varphi^d$  **unfolding** *dual-def NOR-def dNOR-def top-def bottom-def compl-def* **by** *simp*

**lemma** *NOR-dual2*: *dNOR*  $\varphi = \text{NOR } \varphi^d$  **unfolding** *dual-def NOR-def dNOR-def top-def bottom-def compl-def* **by** *simp*

Distribution over meets or multiplicativity (MULT).

**definition** *MULT*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \approx (\varphi A) \wedge (\varphi B)$

**definition** *MULT-a*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \preceq (\varphi A) \wedge (\varphi B)$

**definition** *MULT-b*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \succeq (\varphi A) \wedge (\varphi B)$

Distribution over joins or additivity (ADDI).

**definition** *ADDI*  $\varphi \equiv \forall A B. \varphi(A \vee B) \approx (\varphi A) \vee (\varphi B)$

**definition** *ADDI-a*  $\varphi \equiv \forall A B. \varphi(A \vee B) \preceq (\varphi A) \vee (\varphi B)$

**definition** *ADDI-b*  $\varphi \equiv \forall A B. \varphi(A \vee B) \succeq (\varphi A) \vee (\varphi B)$

## 2.2 Relations among conditions (finitary case)

dEXP and dNOR entail NOR.

**lemma** *dEXP*  $\varphi \implies \text{dNOR } \varphi \implies \text{NOR } \varphi$  **by** (*meson bottom-def dEXP-def NOR-def*)

EXP and NOR entail dNOR.

**lemma** *EXP*  $\varphi \implies \text{NOR } \varphi \implies \text{dNOR } \varphi$  **by** (*simp add: EXP-def dNOR-def top-def*)

Interestingly, EXP and its dual allow for an alternative characterization of fixed-point operators.

**lemma** *EXP-fp*:  $EXP \varphi \implies \varphi^{fp} \equiv (\varphi^c \sqcup id)$  **by** (*smt id-def EXP-def dual-def dual-symm equal-op-def conn*)

**lemma** *dEXP-fp*:  $dEXP \varphi \implies \varphi^{fp} \equiv (\varphi \sqcup compl)$  **by** (*smt dEXP-def equal-op-def conn*)

MONO, MULT-a and ADDI-b are equivalent.

**lemma** *MONO-MULTa*:  $MONO \varphi = MULT\text{-}a \varphi$  **proof** –

**have** *lr*:  $MONO \varphi \implies MULT\text{-}a \varphi$  **by** (*smt MONO-def MULT-a-def meet-def*)

**have** *rl*:  $MULT\text{-}a \varphi \implies MONO \varphi$  **proof** –

**assume** *multa*:  $MULT\text{-}a \varphi$

{ **fix** *A B*

{ **assume**  $A \preceq B$

**hence**  $A \approx A \wedge B$  **unfolding** *conn* **by** *blast*

**hence**  $\varphi A \approx \varphi(A \wedge B)$  **unfolding** *conn* **by** *simp*

**moreover from** *multa* **have**  $\varphi(A \wedge B) \preceq (\varphi A) \wedge (\varphi B)$  **using** *MULT-a-def* **by** *metis*

**ultimately have**  $\varphi A \preceq (\varphi A) \wedge (\varphi B)$  **by** *blast*

**hence**  $\varphi A \preceq (\varphi B)$  **unfolding** *conn* **by** *blast*

} **hence**  $A \preceq B \longrightarrow \varphi A \preceq \varphi B$  **by** (*rule impI*)

} **thus** *?thesis* **by** (*simp add: MONO-def*) **qed**

**from** *lr rl* **show** *?thesis* **by** *auto*

**qed**

**lemma** *MONO-ADDIb*:  $MONO \varphi = ADDI\text{-}b \varphi$  **proof** –

**have** *lr*:  $MONO \varphi \implies ADDI\text{-}b \varphi$  **by** (*smt ADDI-b-def MONO-def join-def*)

**have** *rl*:  $ADDI\text{-}b \varphi \implies MONO \varphi$  **proof** –

**assume** *addib*:  $ADDI\text{-}b \varphi$

{ **fix** *A B*

{ **assume**  $A \preceq B$

**hence**  $B \approx A \vee B$  **unfolding** *conn* **by** *blast*

**hence**  $\varphi B \approx \varphi(A \vee B)$  **unfolding** *conn* **by** *simp*

**moreover from** *addib* **have**  $(\varphi A) \vee (\varphi B) \preceq \varphi(A \vee B)$  **using** *ADDI-b-def* **by** *metis*

**ultimately have**  $(\varphi A) \vee (\varphi B) \preceq \varphi B$  **by** *blast*

**hence**  $\varphi A \preceq (\varphi B)$  **unfolding** *conn* **by** *blast*

} **hence**  $A \preceq B \longrightarrow \varphi A \preceq \varphi B$  **by** (*rule impI*)

} **thus** *?thesis* **by** (*simp add: MONO-def*) **qed**

**from** *lr rl* **show** *?thesis* **by** *auto*

**qed**

**lemma** *ADDIb-MULTa*:  $ADDI\text{-}b \varphi = MULT\text{-}a \varphi$  **using** *MONO-ADDIb MONO-MULTa* **by** *auto*

**end**

**theory** *sse-operation-positive-quantification*

**imports** *sse-operation-positive sse-boolean-algebra-quantification*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 2.3 Definitions (infinitary case)

We define and interrelate infinitary variants for some previously introduced ('positive') conditions on operations and show how they relate to quantifiers as previously defined.

Distribution over infinite meets (infima) or infinite multiplicativity (iMULT).

**definition** *iMULT*  $\varphi \equiv \forall S. \varphi(\bigwedge S) \approx \bigwedge Ra[\varphi]S$

**definition** *iMULT-a*  $\varphi \equiv \forall S. \varphi(\bigwedge S) \preceq \bigwedge Ra[\varphi]S$

**definition** *iMULT-b*  $\varphi \equiv \forall S. \varphi(\bigwedge S) \succeq \bigwedge Ra[\varphi]S$

Distribution over infinite joins (suprema) or infinite additivity (iADDI).

**definition**  $iADDI \varphi \equiv \forall S. \varphi(\bigvee S) \approx \bigvee Ra[\varphi|S]$   
**definition**  $iADDI\text{-}a \varphi \equiv \forall S. \varphi(\bigvee S) \preceq \bigvee Ra[\varphi|S]$   
**definition**  $iADDI\text{-}b \varphi \equiv \forall S. \varphi(\bigvee S) \succeq \bigvee Ra[\varphi|S]$

## 2.4 Relations among conditions (infinitary case)

We start by noting that there is a duality between  $iADDI\text{-}a$  and  $iMULT\text{-}b$ .

**lemma**  $iADDI\text{-}MULT\text{-}dual1: iADDI\text{-}a \varphi \implies iMULT\text{-}b \varphi^d$  **unfolding**  $iADDI\text{-}a\text{-}def$   $iMULT\text{-}b\text{-}def$  **by**  $(metis\ compl\text{-}def\ dual\text{-}def\ iDM\text{-}a\ iDM\text{-}b\ Ra\text{-}dual1)$

**lemma**  $iADDI\text{-}MULT\text{-}dual2: iMULT\text{-}b \varphi \implies iADDI\text{-}a \varphi^d$  **unfolding**  $iADDI\text{-}a\text{-}def$   $iMULT\text{-}b\text{-}def$  **by**  $(metis\ compl\text{-}def\ dual\text{-}def\ iDM\text{-}b\ Ra\text{-}dual3)$

$MULT\text{-}a$  and  $iMULT\text{-}a$  are equivalent.

**lemma**  $iMULTa\text{-}rel: iMULT\text{-}a \varphi = MULT\text{-}a \varphi$  **proof** –

**have**  $lr: iMULT\text{-}a \varphi \implies MULT\text{-}a \varphi$  **proof** –

**assume**  $imulta: iMULT\text{-}a \varphi$

{ **fix**  $A::\sigma$  **and**  $B::\sigma$

**let**  $?S=\lambda Z. Z=A \vee Z=B$

**from**  $imulta$  **have**  $\varphi(\bigwedge ?S) \preceq \bigvee^R(\bigvee ?S) \varphi$  **by**  $(simp\ add: iMULT\text{-}a\text{-}def\ Ra\text{-}restr\text{-}all)$

**moreover** **have**  $\bigwedge ?S = A \wedge B$  **using**  $infimum\text{-}def\ meet\text{-}def$  **by**  $auto$

**moreover** **have**  $\bigvee^R(\bigvee ?S) \varphi = (\varphi A) \wedge (\varphi B)$  **using**  $meet\text{-}def$  **by**  $auto$

**ultimately** **have**  $\varphi(A \wedge B) \preceq (\varphi A) \wedge (\varphi B)$  **by**  $smt$

} **thus**  $?thesis$  **by**  $(simp\ add: MULT\text{-}a\text{-}def)$  **qed**

**have**  $rl: MULT\text{-}a \varphi \implies iMULT\text{-}a \varphi$  **by**  $(smt\ MONO\text{-}def\ MONO\text{-}MULTa\ Ra\text{-}restr\text{-}all\ iMULT\text{-}a\text{-}def\ inf\text{-}char)$

**from**  $lr\ rl$  **show**  $?thesis$  **by**  $auto$

**qed**

$ADDI\text{-}b$  and  $iADDI\text{-}b$  are equivalent.

**lemma**  $iADDIb\text{-}rel: iADDI\text{-}b \varphi = ADDI\text{-}b \varphi$  **proof** –

**have**  $lr: iADDI\text{-}b \varphi \implies ADDI\text{-}b \varphi$  **proof** –

**assume**  $iaddib: iADDI\text{-}b \varphi$

{ **fix**  $A::\sigma$  **and**  $B::\sigma$

**let**  $?S=\lambda Z. Z=A \vee Z=B$

**from**  $iaddib$  **have**  $\varphi(\bigvee ?S) \succeq \bigvee^R(\bigvee ?S)(\varphi)$  **by**  $(simp\ add: iADDI\text{-}b\text{-}def\ Ra\text{-}restr\text{-}ex)$

**moreover** **have**  $\bigvee ?S = A \vee B$  **using**  $supremum\text{-}def\ join\text{-}def$  **by**  $auto$

**moreover** **have**  $\bigvee^R(\bigvee ?S)(\varphi) = (\varphi A) \vee (\varphi B)$  **using**  $join\text{-}def$  **by**  $auto$

**ultimately** **have**  $\varphi(A \vee B) \succeq (\varphi A) \vee (\varphi B)$  **by**  $smt$

} **thus**  $?thesis$  **by**  $(simp\ add: ADDI\text{-}b\text{-}def)$  **qed**

**have**  $rl: ADDI\text{-}b \varphi \implies iADDI\text{-}b \varphi$  **by**  $(smt\ MONO\text{-}def\ MONO\text{-}ADDIb\ Ra\text{-}restr\text{-}ex\ iADDI\text{-}b\text{-}def\ sup\text{-}char)$

**from**  $lr\ rl$  **show**  $?thesis$  **by**  $auto$

**qed**

Thus we have that  $MONO$ ,  $MULT\text{-}a/iMULT\text{-}a$  and  $ADDI\text{-}b/iADDI\text{-}b$  are all equivalent.

**lemma**  $MONO\text{-}iADDIb: MONO \varphi = iADDI\text{-}b \varphi$  **using**  $MONO\text{-}ADDIb\ iADDIb\text{-}rel$  **by**  $simp$

**lemma**  $MONO\text{-}iMULTa: MONO \varphi = iMULT\text{-}a \varphi$  **using**  $MONO\text{-}MULTa\ iMULTa\text{-}rel$  **by**  $simp$

**lemma**  $iADDI\text{-}b\text{-}iMULTa: iADDI\text{-}b \varphi = iMULT\text{-}a \varphi$  **using**  $MONO\text{-}iADDIb\ MONO\text{-}iMULTa$  **by**  $auto$

**lemma**  $PI\text{-}imult: MONO \varphi \implies iMULT\text{-}b \varphi \implies iMULT \varphi$  **using**  $MONO\text{-}MULTa\ iMULT\text{-}a\text{-}def\ iMULT\text{-}b\text{-}def\ iMULT\text{-}def\ iMULTa\text{-}rel$  **by**  $auto$

**lemma**  $PC\text{-}iaddi: MONO \varphi \implies iADDI\text{-}a \varphi \implies iADDI \varphi$  **using**  $MONO\text{-}ADDIb\ iADDI\text{-}a\text{-}def\ iADDI\text{-}b\text{-}def\ iADDI\text{-}def\ iADDIb\text{-}rel$  **by**  $auto$

Interestingly, we can show that suitable (infinitary) conditions on an operation can make the set of its fixed points closed under infinite meets/joins.

**lemma** *fp-inf-closed*:  $MONO \varphi \implies iMULT\text{-}b \varphi \implies \text{infimum-closed } (fp \varphi)$  **by** (*metis (full-types) PI-impl Ra-restr-all iMULT-def infimum-def*)

**lemma** *fp-sup-closed*:  $MONO \varphi \implies iADDI\text{-}a \varphi \implies \text{supremum-closed } (fp \varphi)$  **by** (*metis (full-types) PC-iaddi Ra-restr-ex iADDI-def supremum-def*)

## 2.5 Exploring the Barcan formula and its converse

The converse Barcan formula follows readily from monotonicity.

**lemma** *CBarcan1*:  $MONO \varphi \implies \forall \pi. \varphi(\forall x. \pi x) \preceq (\forall x. \varphi(\pi x))$  **by** (*metis (mono-tags, lifting) MONO-def*)

**lemma** *CBarcan2*:  $MONO \varphi \implies \forall \pi. (\exists x. \varphi(\pi x)) \preceq \varphi(\exists x. \pi x)$  **by** (*metis (mono-tags, lifting) MONO-def*)

However, the Barcan formula requires a stronger assumption (of an infinitary character).

**lemma** *Barcan1*:  $iMULT\text{-}b \varphi \implies \forall \pi. (\forall x. \varphi(\pi x)) \preceq \varphi(\forall x. \pi x)$  **proof** –

**assume** *imultb*:  $iMULT\text{-}b \varphi$

**{ fix**  $\pi::'a \Rightarrow \sigma$

**from** *imultb* **have**  $(\bigwedge Ra(\varphi \circ \pi)) \preceq \varphi(\bigwedge Ra(\pi))$  **unfolding** *iMULT-b-def* **by** (*smt comp-apply infimum-def pfunRange-def pfunRange-restr-def*)

**moreover** **have**  $\bigwedge Ra(\pi) = (\forall x. \pi x)$  **unfolding** *Ra-all* **by** *simp*

**moreover** **have**  $\bigwedge Ra(\varphi \circ \pi) = (\forall x. \varphi(\pi x))$  **unfolding** *Ra-all* **by** *simp*

**ultimately** **have**  $(\forall x. \varphi(\pi x)) \preceq \varphi(\forall x. \pi x)$  **by** *simp*

**} thus** *?thesis* **by** *simp*

**qed**

**lemma** *Barcan2*:  $iADDI\text{-}a \varphi \implies \forall \pi. \varphi(\exists x. \pi x) \preceq (\exists x. \varphi(\pi x))$  **proof** –

**assume** *iaddia*:  $iADDI\text{-}a \varphi$

**{ fix**  $\pi::'a \Rightarrow \sigma$

**from** *iaddia* **have**  $\varphi(\bigvee Ra(\pi)) \preceq (\bigvee Ra(\varphi \circ \pi))$  **unfolding** *iADDI-a-def Ra-restr-ex* **by** (*smt fcomp-comp fcomp-def pfunRange-def sup-char*)

**moreover** **have**  $\bigvee Ra(\pi) = (\exists x. \pi x)$  **unfolding** *Ra-ex* **by** *simp*

**moreover** **have**  $\bigvee Ra(\varphi \circ \pi) = (\exists x. \varphi(\pi x))$  **unfolding** *Ra-ex* **by** *simp*

**ultimately** **have**  $\varphi(\exists x. \pi x) \preceq (\exists x. \varphi(\pi x))$  **by** *simp*

**} thus** *?thesis* **by** *simp*

**qed**

**end**

**theory** *sse-operation-negative*

**imports** *sse-boolean-algebra*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 3 Negative semantic conditions for operations

We define and interrelate some conditions on operations (i.e. propositional functions of type  $\sigma \Rightarrow \sigma$ ), this time involving negative-like properties.

**named-theorems** *Defs*

### 3.1 Definitions and interrelations (finitary case)

#### 3.1.1 Principles of excluded middle, contradiction and explosion

TND: tertium non datur, aka. law of excluded middle (resp. strong, weak, minimal).

**abbreviation**  $pTND$  ( $TND^-$  -) **where**  $TND^a \eta \equiv \top \approx a \vee (\eta a)$

**abbreviation**  $pTNDw$  ( $TNDw^-$  -) **where**  $TNDw^a \eta \equiv \forall b. (\eta b) \preceq a \vee (\eta a)$

**abbreviation**  $pTNDm$  ( $TNDm^-$  -) **where**  $TNDm^a \eta \equiv (\eta \perp) \preceq a \vee (\eta a)$

**definition**  $TND \eta \equiv \forall \varphi. TND^\varphi \eta$

**definition**  $TNDw \eta \equiv \forall \varphi. TNDw^\varphi \eta$

**definition**  $TNDm \eta \equiv \forall \varphi. TNDm^\varphi \eta$

**declare**  $TND\text{-def}[Defs]$   $TNDw\text{-def}[Defs]$   $TNDm\text{-def}[Defs]$

Explore some (non)entailment relations:

**lemma**  $TND \eta \implies TNDw \eta$  **unfolding**  $Defs$  **conn** **by**  $simp$

**lemma**  $TNDw \eta \implies TND \eta$  **nitpick** **oops**

**lemma**  $TNDw \eta \implies TNDm \eta$  **unfolding**  $Defs$  **by**  $simp$

**lemma**  $TNDm \eta \implies TNDw \eta$  **nitpick** **oops**

ECQ: ex contradictione (sequitur) quodlibet (resp: strong, weak, minimal).

**abbreviation**  $pECQ$  ( $ECQ^-$  -) **where**  $ECQ^a \eta \equiv a \wedge (\eta a) \approx \perp$

**abbreviation**  $pECQw$  ( $ECQw^-$  -) **where**  $ECQw^a \eta \equiv \forall b. a \wedge (\eta a) \preceq (\eta b)$

**abbreviation**  $pECQm$  ( $ECQm^-$  -) **where**  $ECQm^a \eta \equiv a \wedge (\eta a) \preceq (\eta \top)$

**definition**  $ECQ \eta \equiv \forall a. ECQ^a \eta$

**definition**  $ECQw \eta \equiv \forall a. ECQw^a \eta$

**definition**  $ECQm \eta \equiv \forall a. ECQm^a \eta$

**declare**  $ECQ\text{-def}[Defs]$   $ECQw\text{-def}[Defs]$   $ECQm\text{-def}[Defs]$

Explore some (non)entailment relations:

**lemma**  $ECQ \eta \implies ECQw \eta$  **unfolding**  $Defs$  **conn** **by**  $blast$

**lemma**  $ECQw \eta \implies ECQ \eta$  **nitpick** **oops**

**lemma**  $ECQw \eta \implies ECQm \eta$  **unfolding**  $Defs$  **conn** **by**  $simp$

**lemma**  $ECQm \eta \implies ECQw \eta$  **nitpick** **oops**

LNC: law of non-contradiction.

**abbreviation**  $pLNC$  ( $LNC^-$  -) **where**  $LNC^a \eta \equiv \eta(a \wedge \eta a) \approx \top$

**definition**  $LNC \eta \equiv \forall a. LNC^a \eta$

**declare**  $LNC\text{-def}[Defs]$

ECQ and LNC are in general independent.

**lemma**  $ECQ \eta \implies LNC \eta$  **nitpick** **oops**

**lemma**  $LNC \eta \implies ECQm \eta$  **nitpick** **oops**

#### 3.1.2 Contraposition rules

CoP: contraposition (global/rule variants, resp. weak, strong var. 1, strong var. 2, strong var. 3).

**abbreviation**  $pCoPw$  ( $CoPw^{--}$  -) **where**  $CoPw^{ab} \eta \equiv a \preceq b \longrightarrow (\eta b) \preceq (\eta a)$

**abbreviation**  $pCoP1$  ( $CoP1^{--}$  -) **where**  $CoP1^{ab} \eta \equiv a \preceq (\eta b) \longrightarrow b \preceq (\eta a)$

**abbreviation**  $pCoP2$  ( $CoP2^{--}$  -) **where**  $CoP2^{ab} \eta \equiv (\eta a) \preceq b \longrightarrow (\eta b) \preceq a$

**abbreviation**  $pCoP3$  ( $CoP3^{--}$  -) **where**  $CoP3^{ab} \eta \equiv (\eta a) \preceq (\eta b) \longrightarrow b \preceq a$

**definition**  $CoPw \eta \equiv \forall a b. CoPw^{ab} \eta$

**definition**  $CoP1 \eta \equiv \forall a b. CoP1^{ab} \eta$

**definition**  $CoP1' \eta \equiv \forall a b. a \preceq (\eta b) \longleftrightarrow b \preceq (\eta a)$

**definition**  $CoP2 \eta \equiv \forall a b. CoP2^{ab} \eta$

**definition**  $CoP2' \eta \equiv \forall a b. (\eta a) \preceq b \longleftrightarrow (\eta b) \preceq a$

**definition**  $CoP3 \eta \equiv \forall a b. CoP3^{ab} \eta$

**declare**  $CoPw-def[Defs]$   $CoP1-def[Defs]$   $CoP1'-def[Defs]$   
 $CoP2-def[Defs]$   $CoP2'-def[Defs]$   $CoP3-def[Defs]$

**lemma**  $CoP1-defs-rel: CoP1 \eta = CoP1' \eta$  **unfolding**  $Defs$  **by** *metis*

**lemma**  $CoP2-defs-rel: CoP2 \eta = CoP2' \eta$  **unfolding**  $Defs$  **by** *metis*

Explore some (non)entailment relations:

**lemma**  $CoP1 \eta \implies CoPw \eta$  **unfolding**  $Defs$  **by** *metis*

**lemma**  $CoPw \eta \implies CoP1 \eta$  **nitpick oops**

**lemma**  $CoP2 \eta \implies CoPw \eta$  **unfolding**  $Defs$  **by** *metis*

**lemma**  $CoPw \eta \implies CoP2 \eta$  **nitpick oops**

**lemma**  $CoP3 \eta \implies CoPw \eta$  **oops** — no countermodel found so far

**lemma**  $CoPw \eta \implies CoP3 \eta$  **nitpick oops**

All three strong variants are pairwise independent. However,  $CoP3$  follows from  $CoP1$  plus  $CoP2$ .

**lemma**  $CoP123: CoP1 \eta \implies CoP2 \eta \implies CoP3 \eta$  **unfolding**  $Defs$  **by** *smt*

Taking all  $CoP$  together still leaves room for a boldly paraconsistent resp. paracomplete logic.

**lemma**  $CoP1 \eta \implies CoP2 \eta \implies ECQm \eta$  **nitpick oops**

**lemma**  $CoP1 \eta \implies CoP2 \eta \implies TNDm \eta$  **nitpick oops**

### 3.1.3 Modus tollens rules

MT: modus (tollendo) tollens (global/rule variants).

**abbreviation**  $pMT0$  ( $MT0^{--}$  -) **where**  $MT0^{ab} \eta \equiv a \preceq b \wedge (\eta b) \approx \top \longrightarrow (\eta a) \approx \top$

**abbreviation**  $pMT1$  ( $MT1^{--}$  -) **where**  $MT1^{ab} \eta \equiv a \preceq (\eta b) \wedge b \approx \top \longrightarrow (\eta a) \approx \top$

**abbreviation**  $pMT2$  ( $MT2^{--}$  -) **where**  $MT2^{ab} \eta \equiv (\eta a) \preceq b \wedge (\eta b) \approx \top \longrightarrow a \approx \top$

**abbreviation**  $pMT3$  ( $MT3^{--}$  -) **where**  $MT3^{ab} \eta \equiv (\eta a) \preceq (\eta b) \wedge b \approx \top \longrightarrow a \approx \top$

**definition**  $MT0 \eta \equiv \forall a b. MT0^{ab} \eta$

**definition**  $MT1 \eta \equiv \forall a b. MT1^{ab} \eta$

**definition**  $MT2 \eta \equiv \forall a b. MT2^{ab} \eta$

**definition**  $MT3 \eta \equiv \forall a b. MT3^{ab} \eta$

**declare**  $MT0-def[Defs]$   $MT1-def[Defs]$   $MT2-def[Defs]$   $MT3-def[Defs]$

Again, all MT variants are pairwise independent. We explore some (non)entailment relations:

**lemma**  $CoPw \eta \implies MT0 \eta$  **unfolding**  $Defs$  **by** (*metis top-def*)

**lemma**  $CoP1 \eta \implies MT1 \eta$  **unfolding**  $Defs$  **by** (*metis top-def*)

**lemma**  $CoP2 \eta \implies MT2 \eta$  **unfolding**  $Defs$  **by** (*metis top-def*)

**lemma**  $CoP3 \eta \implies MT3 \eta$  **unfolding**  $Defs$  **by** (*metis top-def*)

**lemma**  $MT0 \eta \implies MT1 \eta \implies MT2 \eta \implies MT3 \eta \implies CoPw \eta$  **nitpick oops**

**lemma**  $MT0 \eta \implies MT1 \eta \implies MT2 \eta \implies MT3 \eta \implies ECQm \eta$  **nitpick oops**

**lemma**  $MT0 \eta \implies MT1 \eta \implies MT2 \eta \implies MT3 \eta \implies TNDm \eta$  **nitpick oops**

**lemma**  $MT123: MT1 \eta \implies MT2 \eta \implies MT3 \eta$  **unfolding**  $Defs$  **by** *smt*

### 3.1.4 Double negation introduction and elimination

DNI/DNE: double negation introduction/elimination (as axioms).

**abbreviation**  $pDNI$  ( $DNI^-$  -) **where**  $DNI^a \eta \equiv a \preceq \eta (\eta a)$

**abbreviation**  $pDNE$  ( $DNE^-$  -) **where**  $DNE^a \eta \equiv \eta (\eta a) \preceq a$

**definition**  $DNI \eta \equiv \forall a. DNI^a \eta$

**definition**  $DNE \eta \equiv \forall a. DNE^a \eta$

**declare**  $DNI-def[Defs]$   $DNE-def[Defs]$

CoP1 (resp. CoP2) can alternatively be defined as CoPw plus DNI (resp. DNE).

**lemma**  $DNI \eta \implies CoP1 \eta$  **nitpick oops**

**lemma**  $CoP1-def2: CoP1 \eta = (CoPw \eta \wedge DNI \eta)$  **unfolding**  $Defs$  **by**  $smt$

**lemma**  $DNE \eta \implies CoP2 \eta$  **nitpick oops**

**lemma**  $CoP2-def2: CoP2 \eta = (CoPw \eta \wedge DNE \eta)$  **unfolding**  $Defs$  **by**  $smt$

Explore some non-entailment relations:

**lemma**  $DNI \eta \implies DNE \eta \implies CoPw \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies TNDm \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies ECQm \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies MT0 \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies MT1 \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies MT2 \eta$  **nitpick oops**

**lemma**  $DNI \eta \implies DNE \eta \implies MT3 \eta$  **nitpick oops**

DNI/DNE: double negation introduction/elimination (as rules).

**abbreviation**  $prDNI$  ( $rDNI^-$  -) **where**  $rDNI^a \eta \equiv a \approx \top \longrightarrow \eta (\eta a) \approx \top$

**abbreviation**  $prDNE$  ( $rDNE^-$  -) **where**  $rDNE^a \eta \equiv \eta (\eta a) \approx \top \longrightarrow a \approx \top$

**definition**  $rDNI \eta \equiv \forall a. rDNI^a \eta$

**definition**  $rDNE \eta \equiv \forall a. rDNE^a \eta$

**declare**  $rDNI-def[Defs]$   $rDNE-def[Defs]$

The rule variants are strictly weaker than the axiom variants,

**lemma**  $DNI \eta \implies rDNI \eta$  **by** (*simp add: DNI-def rDNI-def top-def*)

**lemma**  $rDNI \eta \implies DNI \eta$  **nitpick oops**

**lemma**  $DNE \eta \implies rDNE \eta$  **by** (*metis DNE-def rDNE-def top-def*)

**lemma**  $rDNE \eta \implies DNE \eta$  **nitpick oops**

and follow already from modus tollens.

**lemma**  $MT1-rDNI: MT1 \eta \implies rDNI \eta$  **unfolding**  $Defs$  **by**  $blast$

**lemma**  $MT2-rDNE: MT2 \eta \implies rDNE \eta$  **unfolding**  $Defs$  **by**  $blast$

### 3.1.5 Normality and its dual

n(D)Nor: negative (dual) 'normality'.

**definition**  $nNor \eta \equiv (\eta \perp) \approx \top$

**definition**  $nDNor \eta \equiv (\eta \top) \approx \perp$

**declare**  $nNor-def[Defs]$   $nDNor-def[Defs]$

nNor (resp. nDNor) is entailed by CoP1 (resp. CoP2).

**lemma**  $CoP1-Nor: CoP1 \eta \implies nNor \eta$  **unfolding**  $Defs$  **conn** **by**  $simp$

**lemma**  $CoP2-DNor: CoP2 \eta \implies nDNor \eta$  **unfolding**  $Defs$  **conn** **by**  $fastforce$

**lemma**  $DNI \eta \implies nNor \eta$  **nitpick oops**

**lemma**  $DNE \eta \implies nDNor \eta$  **nitpick oops**

nNor and nDNor together entail the rule variant of DNI (rDNI).

**lemma**  $nDNor-rDNI: nNor \eta \implies nDNor \eta \implies rDNI \eta$  **unfolding**  $Defs$  **using**  $nDNor-def$   $nNor-def$  *eq-ext* **by**  $metis$

**lemma**  $nNor \eta \implies nDNor \eta \implies rDNE \eta$  **nitpick oops**



### 3.1.6 De Morgan laws

DM: De Morgan laws.

**abbreviation**  $\overline{pDM1}$  ( $DM1^{--}$  -) **where**  $DM1^{ab} \eta \equiv \eta(a \vee b) \preceq (\eta a) \wedge (\eta b)$   
**abbreviation**  $pDM2$  ( $DM2^{--}$  -) **where**  $DM2^{ab} \eta \equiv (\eta a) \vee (\eta b) \preceq \eta(a \wedge b)$   
**abbreviation**  $\overline{pDM3}$  ( $DM3^{--}$  -) **where**  $DM3^{ab} \eta \equiv \eta(a \wedge b) \preceq (\eta a) \vee (\eta b)$   
**abbreviation**  $pDM4$  ( $DM4^{--}$  -) **where**  $DM4^{ab} \eta \equiv (\eta a) \wedge (\eta b) \preceq \eta(a \vee b)$   
**definition**  $DM1 \eta \equiv \forall a b. DM1^{ab} \eta$   
**definition**  $DM2 \eta \equiv \forall a b. DM2^{ab} \eta$   
**definition**  $DM3 \eta \equiv \forall a b. DM3^{ab} \eta$   
**definition**  $DM4 \eta \equiv \forall a b. DM4^{ab} \eta$   
**declare**  $DM1\text{-def}[Defs]$   $DM2\text{-def}[Defs]$   $DM3\text{-def}[Defs]$   $DM4\text{-def}[Defs]$

CoPw, DM1 and DM2 are indeed equivalent.

**lemma**  $DM1\text{-CoPw}$ :  $DM1 \eta = CoPw \eta$  **proof** -  
**have**  $LtoR$ :  $DM1 \eta \implies CoPw \eta$  **proof** -  
**assume**  $dm1$ :  $DM1 \eta$   
{ **fix**  $a b$   
{ **assume**  $a \preceq b$   
**hence**  $1$ :  $a \vee b \preceq b$  **unfolding** *conn* **by** *simp*  
**have**  $2$ :  $b \preceq a \vee b$  **unfolding** *conn* **by** *simp*  
**from**  $1\ 2$  **have**  $b = a \vee b$  **using** *eq-ext* **by** *blast*  
**hence**  $3$ :  $\eta b \approx \eta(a \vee b)$  **by** *auto*  
**from**  $dm1$  **have**  $\eta(a \vee b) \preceq (\eta a) \wedge (\eta b)$  **unfolding** *Defs* **by** *blast*  
**hence**  $4$ :  $(\eta b) \preceq (\eta a) \wedge (\eta b)$  **using**  $3$  **by** *simp*  
**have**  $5$ :  $(\eta a) \wedge (\eta b) \preceq (\eta a)$  **unfolding** *conn* **by** *simp*  
**from**  $4\ 5$  **have**  $(\eta b) \preceq (\eta a)$  **by** *simp*  
} **hence**  $a \preceq b \longrightarrow (\eta b) \preceq (\eta a)$  **by** (*rule impI*)  
} **thus** *?thesis* **unfolding** *Defs* **by** *simp*  
**qed**  
**have**  $RtoL$ :  $CoPw \eta \implies DM1 \eta$  **unfolding** *Defs* *conn* **by** (*metis* (*no-types*, *lifting*))  
**thus** *?thesis* **using**  $LtoR$   $RtoL$  **by** *blast*  
**qed**  
**lemma**  $DM2\text{-CoPw}$ :  $DM2 \eta = CoPw \eta$  **proof** -  
**have**  $LtoR$ :  $DM2 \eta \implies CoPw \eta$  **proof** -  
**assume**  $dm2$ :  $DM2 \eta$   
{ **fix**  $a b$   
{ **assume**  $a \preceq b$   
**hence**  $1$ :  $a \preceq a \wedge b$  **unfolding** *conn* **by** *simp*  
**have**  $2$ :  $a \wedge b \preceq a$  **unfolding** *conn* **by** *simp*  
**from**  $1\ 2$  **have**  $a = a \wedge b$  **using** *eq-ext* **by** *blast*  
**hence**  $3$ :  $\eta a \approx \eta(a \wedge b)$  **by** *auto*  
**from**  $dm2$  **have**  $(\eta a) \vee (\eta b) \preceq \eta(a \wedge b)$  **unfolding** *Defs* **by** *blast*  
**hence**  $4$ :  $(\eta a) \vee (\eta b) \preceq (\eta a)$  **using**  $3$  **by** *simp*  
**have**  $5$ :  $(\eta b) \preceq (\eta a) \vee (\eta b)$  **unfolding** *conn* **by** *simp*  
**from**  $4\ 5$  **have**  $(\eta b) \preceq (\eta a)$  **by** *simp*  
} **hence**  $a \preceq b \longrightarrow (\eta b) \preceq (\eta a)$  **by** (*rule impI*)  
} **thus** *?thesis* **unfolding** *Defs* **by** *simp*  
**qed**  
**have**  $RtoL$ :  $CoPw \eta \implies DM2 \eta$  **unfolding** *Defs* *conn* **by** (*metis* (*no-types*, *lifting*))  
**thus** *?thesis* **using**  $LtoR$   $RtoL$  **by** *blast*  
**qed**

**lemma**  $DM12$ :  $DM1 \eta = DM2 \eta$  **by** (*simp add*:  $DM1\text{-CoPw}$   $DM2\text{-CoPw}$ )

DM3 (resp. DM4) are entailed by CoPw together with DNE (resp. DNI).

**lemma** *CoPw-DNE-DM3*:  $CoPw\ \eta \implies DNE\ \eta \implies DM3\ \eta$  **proof** –

**assume** *copw*:  $CoPw\ \eta$  **and** *dne*:  $DNE\ \eta$

{ **fix**  $a\ b$

**have**  $\eta(a) \preceq (\eta\ a) \vee (\eta\ b)$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(\eta(a) \vee \eta(b)) \preceq \eta((\eta\ a))$  **using** *CoPw-def copw* **by** (*metis* (*no-types*, *lifting*))

**hence**  $1: \eta(\eta(a) \vee \eta(b)) \preceq a$  **using** *DNE-def dne* **by** *metis*

**have**  $\eta(b) \preceq (\eta\ a) \vee (\eta\ b)$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(\eta(a) \vee \eta(b)) \preceq \eta((\eta\ b))$  **using** *CoPw-def copw* **by** (*metis* (*no-types*, *lifting*))

**hence**  $2: \eta(\eta(a) \vee \eta(b)) \preceq b$  **using** *DNE-def dne* **by** *metis*

**from**  $1\ 2$  **have**  $\eta(\eta(a) \vee \eta(b)) \preceq a \wedge b$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(a \wedge b) \preceq \eta(\eta(\eta(a) \vee \eta(b)))$  **using** *CoPw-def copw* **by** *smt*

**hence**  $\eta(a \wedge b) \preceq (\eta\ a) \vee (\eta\ b)$  **using** *DNE-def dne* **by** *metis*

} **thus** *?thesis* **by** (*simp add: DM3-def*)

**qed**

**lemma** *CoPw-DNI-DM4*:  $CoPw\ \eta \implies DNI\ \eta \implies DM4\ \eta$  **proof** –

**assume** *copw*:  $CoPw\ \eta$  **and** *dni*:  $DNI\ \eta$

{ **fix**  $a\ b$

**have**  $(\eta\ a) \wedge (\eta\ b) \preceq \eta(a)$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(\eta\ a) \preceq \eta(\eta(a) \wedge \eta(b))$  **using** *CoPw-def copw* **by** (*metis* (*no-types*, *lifting*))

**hence**  $1: a \preceq \eta(\eta(a) \wedge \eta(b))$  **using** *DNI-def dni* **by** *metis*

**have**  $(\eta\ a) \wedge (\eta\ b) \preceq \eta(b)$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(\eta\ b) \preceq \eta(\eta(a) \wedge \eta(b))$  **using** *CoPw-def copw* **by** (*metis* (*no-types*, *lifting*))

**hence**  $2: b \preceq \eta(\eta(a) \wedge \eta(b))$  **using** *DNI-def dni* **by** *metis*

**from**  $1\ 2$  **have**  $a \vee b \preceq \eta(\eta(a) \wedge \eta(b))$  **unfolding** *conn* **by** *simp*

**hence**  $\eta(\eta(\eta(a) \wedge \eta(b))) \preceq \eta(a \vee b)$  **using** *CoPw-def copw* **by** *auto*

**hence**  $\eta(a) \wedge \eta(b) \preceq \eta(a \vee b)$  **using** *DNI-def dni* **by** *simp*

} **thus** *?thesis* **by** (*simp add: DM4-def*)

**qed**

From this follows that DM3 (resp. DM4) is entailed by CoP2 (resp. CoP1).

**lemma** *CoP2-DM3*:  $CoP2\ \eta \implies DM3\ \eta$  **by** (*simp add: CoP2-def2 CoPw-DNE-DM3*)

**lemma** *CoP1-DM4*:  $CoP1\ \eta \implies DM4\ \eta$  **by** (*simp add: CoP1-def2 CoPw-DNI-DM4*)

Explore some non-entailment relations:

**lemma**  $CoPw\ \eta \implies DM3\ \eta \implies DM4\ \eta \implies nNor\ \eta \implies nDNor\ \eta \implies DNI\ \eta$  **nitpick** **oops**

**lemma**  $CoPw\ \eta \implies DM3\ \eta \implies DM4\ \eta \implies nNor\ \eta \implies nDNor\ \eta \implies DNE\ \eta$  **nitpick** **oops**

**lemma**  $CoPw\ \eta \implies DM3\ \eta \implies DM4\ \eta \implies DNI\ \eta \implies DNE\ \eta \implies ECQm\ \eta$  **nitpick** **oops**

**lemma**  $CoPw\ \eta \implies DM3\ \eta \implies DM4\ \eta \implies DNI\ \eta \implies DNE\ \eta \implies TNDm\ \eta$  **nitpick** **oops**

### 3.1.7 Contextual (strong) contraposition rule

XCoP: contextual contraposition (global/rule variant).

**abbreviation**  $pXCoP\ (XCoP^- -)$  **where**  $XCoP^{ab}\ \eta \equiv \forall c. c \wedge a \preceq b \longrightarrow c \wedge (\eta\ b) \preceq (\eta\ a)$

**definition**  $XCoP\ \eta \equiv \forall a\ b. XCoP^{ab}\ \eta$

**declare**  $XCoP-def[Defs]$

XCoP can alternatively be defined as ECQw plus TNDw.

**lemma** *XCoP-def2*:  $XCoP\ \eta = (ECQw\ \eta \wedge TNDw\ \eta)$  **proof** –

**have** *LtoR1*:  $XCoP\ \eta \implies ECQw\ \eta$  **unfolding** *XCoP-def ECQw-def* *conn* **by** *auto*

**have** *LtoR2*:  $XCoP\ \eta \implies TNDw\ \eta$  **unfolding** *XCoP-def TNDw-def* *conn* **by** (*smt atom-def atomic2 conn*)

**have** *RtoL*:  $ECQw\ \eta \wedge TNDw\ \eta \implies XCoP\ \eta$  **using** *XCoP-def ECQw-def TNDw-def* **unfolding** *conn* **by** *metis*

**from** *LtoR1 LtoR2 RtoL* **show** *?thesis* **by** *blast*

qed

Explore some (non)entailment relations:

**lemma**  $XCoP \eta \implies ECQ \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies TND \eta$  **nitpick oops**

**lemma**  $XCoP-CoPw$ :  $XCoP \eta \implies CoPw \eta$  **unfolding Defs conn by metis**

**lemma**  $XCoP \eta \implies CoP1 \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies CoP2 \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies CoP3 \eta$  **nitpick oops**

**lemma**  $CoP1 \eta \wedge CoP2 \eta \implies XCoP \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies nNor \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies nDNor \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies rDNI \eta$  **nitpick oops**

**lemma**  $XCoP \eta \implies rDNE \eta$  **nitpick oops**

**lemma**  $XCoP-DM3$ :  $XCoP \eta \implies DM3 \eta$  **unfolding DM3-def XCoP-def conn using ECQw-def TNDw-def atom-def atomic2 conn by smt**

**lemma**  $XCoP-DM4$ :  $XCoP \eta \implies DM4 \eta$  **unfolding DM4-def XCoP-def conn using ECQw-def TNDw-def atom-def atomic2 conn by smt**

### 3.1.8 Local contraposition axioms

Observe that the definitions below take implication as an additional parameter:  $\iota$ .

lCoP: contraposition (local/axiom variants).

**abbreviation**  $plCoPw$  ( $lCoPw^{--}$  - -) **where**  $lCoPw^{ab} \iota \eta \equiv (\iota a b::\sigma) \preceq (\iota (\eta b) (\eta a))$

**abbreviation**  $plCoP1$  ( $lCoP1^{--}$  - -) **where**  $lCoP1^{ab} \iota \eta \equiv (\iota a (\eta b::\sigma)) \preceq (\iota b (\eta a))$

**abbreviation**  $plCoP2$  ( $lCoP2^{--}$  - -) **where**  $lCoP2^{ab} \iota \eta \equiv (\iota (\eta a) b::\sigma) \preceq (\iota (\eta b) a)$

**abbreviation**  $plCoP3$  ( $lCoP3^{--}$  - -) **where**  $lCoP3^{ab} \iota \eta \equiv (\iota (\eta a) (\eta b::\sigma)) \preceq (\iota b a)$

**definition**  $lCoPw \iota \eta \equiv \forall a b. lCoPw^{ab} \iota \eta$

**definition**  $lCoP1 \iota \eta \equiv \forall a b. lCoP1^{ab} \iota \eta$

**definition**  $lCoP1' \iota \eta \equiv \forall a b. (\iota a (\eta b)) \approx (\iota b (\eta a))$

**definition**  $lCoP2 \iota \eta \equiv \forall a b. lCoP2^{ab} \iota \eta$

**definition**  $lCoP2' \iota \eta \equiv \forall a b. (\iota (\eta a) b) \approx (\iota (\eta b) a)$

**definition**  $lCoP3 \iota \eta \equiv \forall a b. lCoP3^{ab} \iota \eta$

**declare**  $lCoPw$ -def[Defs]  $lCoP1$ -def[Defs]  $lCoP1'$ -def[Defs]  
 $lCoP2$ -def[Defs]  $lCoP2'$ -def[Defs]  $lCoP3$ -def[Defs]

**lemma**  $lCoP1$ -defs-rel:  $lCoP1 \iota \eta = lCoP1' \iota \eta$  **by** (*metis (full-types) lCoP1'-def lCoP1-def*)

**lemma**  $lCoP2$ -defs-rel:  $lCoP2 \iota \eta = lCoP2' \iota \eta$  **by** (*metis (full-types) lCoP2'-def lCoP2-def*)

All local contraposition variants are in general independent from each other. However if we take classical implication we can verify some relationships.

**lemma**  $lCoP1$ -def2:  $lCoP1(\rightarrow) \eta = (lCoPw(\rightarrow) \eta \wedge DNI \eta)$  **unfolding Defs conn by smt**

**lemma**  $lCoP2$ -def2:  $lCoP2(\rightarrow) \eta = (lCoPw(\rightarrow) \eta \wedge DNE \eta)$  **unfolding Defs conn by smt**

**lemma**  $lCoP1(\rightarrow) \eta \implies lCoPw(\rightarrow) \eta$  **unfolding Defs conn by metis**

**lemma**  $lCoPw(\rightarrow) \eta \implies lCoP1(\rightarrow) \eta$  **nitpick oops**

**lemma**  $lCoP2(\rightarrow) \eta \implies lCoPw(\rightarrow) \eta$  **unfolding Defs conn by metis**

**lemma**  $lCoPw(\rightarrow) \eta \implies lCoP2(\rightarrow) \eta$  **nitpick oops**

**lemma**  $lCoP3(\rightarrow) \eta \implies lCoPw(\rightarrow) \eta$  **unfolding Defs conn by blast**

**lemma**  $lCoPw(\rightarrow) \eta \implies lCoP3(\rightarrow) \eta$  **nitpick oops**

**lemma**  $lCoP123$ :  $lCoP1(\rightarrow) \eta \wedge lCoP2(\rightarrow) \eta \implies lCoP3(\rightarrow) \eta$  **unfolding Defs conn by metis**

Local variants imply global ones as expected.

**lemma**  $lCoPw(\rightarrow) \eta \implies CoPw \eta$  **unfolding Defs conn by metis**

**lemma**  $lCoP1(\rightarrow) \eta \implies CoP1 \eta$  **unfolding** *Defs conn by metis*  
**lemma**  $lCoP2(\rightarrow) \eta \implies CoP2 \eta$  **unfolding** *Defs conn by metis*  
**lemma**  $lCoP3(\rightarrow) \eta \implies CoP3 \eta$  **unfolding** *Defs conn by metis*

Explore some (non)entailment relations.

**lemma**  $lCoPw-XCoP: lCoPw(\rightarrow) \eta = XCoP \eta$  **unfolding** *Defs conn by (smt XCoP-def XCoP-def2 TNDw-def conn)*

**lemma**  $lCoP1-TND: lCoP1(\rightarrow) \eta \implies TND \eta$  **by** *(smt XCoP-CoPw XCoP-def2 CoP1-Nor CoP1-def2 nNor-def TND-def TNDw-def lCoP1-def2 lCoPw-XCoP conn)*

**lemma**  $TND \eta \implies lCoP1(\rightarrow) \eta$  **nitpick oops**

**lemma**  $lCoP2-ECQ: lCoP2(\rightarrow) \eta \implies ECQ \eta$  **by** *(smt XCoP-CoPw XCoP-def2 CoP2-DNor CoP2-def2 nDNor-def ECQ-def ECQw-def lCoP2-def2 lCoPw-XCoP conn)*

**lemma**  $ECQ \eta \implies lCoP2(\rightarrow) \eta$  **nitpick oops**

### 3.1.9 Local modus tollens axioms

IMT: Modus tollens (local/axiom variants).

**abbreviation**  $plMT0$  ( $lMT0^{--}$  - -) **where**  $lMT0^{ab} \iota \eta \equiv (\iota a b::\sigma) \wedge (\eta b) \preceq (\eta a)$

**abbreviation**  $plMT1$  ( $lMT1^{--}$  - -) **where**  $lMT1^{ab} \iota \eta \equiv (\iota a (\eta b::\sigma)) \wedge b \preceq (\eta a)$

**abbreviation**  $plMT2$  ( $lMT2^{--}$  - -) **where**  $lMT2^{ab} \iota \eta \equiv (\iota (\eta a) b::\sigma) \wedge (\eta b) \preceq a$

**abbreviation**  $plMT3$  ( $lMT3^{--}$  - -) **where**  $lMT3^{ab} \iota \eta \equiv (\iota (\eta a) (\eta b::\sigma)) \wedge b \preceq a$

**definition**  $lMT0 \iota \eta \equiv \forall a b. lMT0^{ab} \iota \eta$

**definition**  $lMT1 \iota \eta \equiv \forall a b. lMT1^{ab} \iota \eta$

**definition**  $lMT2 \iota \eta \equiv \forall a b. lMT2^{ab} \iota \eta$

**definition**  $lMT3 \iota \eta \equiv \forall a b. lMT3^{ab} \iota \eta$

**declare**  $lMT0-def[Defs]$   $lMT1-def[Defs]$   $lMT2-def[Defs]$   $lMT3-def[Defs]$

All local MT variants are in general independent from each other and also from local CoP instances. However if we take classical implication we can verify that local MT and CoP are indeed equivalent.

**lemma**  $lMT0(\rightarrow) \eta = lCoPw(\rightarrow) \eta$  **unfolding** *Defs conn by metis*

**lemma**  $lMT1(\rightarrow) \eta = lCoP1(\rightarrow) \eta$  **unfolding** *Defs conn by metis*

**lemma**  $lMT2(\rightarrow) \eta = lCoP2(\rightarrow) \eta$  **unfolding** *Defs conn by metis*

**lemma**  $lMT3(\rightarrow) \eta = lCoP3(\rightarrow) \eta$  **unfolding** *Defs conn by metis*

### 3.1.10 Disjunctive syllogism

DS: disjunctive syllogism.

**abbreviation**  $pDS1$  ( $DS1^{--}$  - -) **where**  $DS1^{ab} \iota \eta \equiv (a \vee b::\sigma) \preceq (\iota (\eta a) b)$

**abbreviation**  $pDS2$  ( $DS2^{--}$  - -) **where**  $DS2^{ab} \iota \eta \equiv (\iota (\eta a) b::\sigma) \preceq (a \vee b)$

**abbreviation**  $pDS3$  ( $DS3^{--}$  - -) **where**  $DS3^{ab} \iota \eta \equiv ((\eta a) \vee b::\sigma) \preceq (\iota a b)$

**abbreviation**  $pDS4$  ( $DS4^{--}$  - -) **where**  $DS4^{ab} \iota \eta \equiv (\iota a b::\sigma) \preceq ((\eta a) \vee b)$

**definition**  $DS1 \iota \eta \equiv \forall a b. DS1^{ab} \iota \eta$

**definition**  $DS2 \iota \eta \equiv \forall a b. DS2^{ab} \iota \eta$

**definition**  $DS3 \iota \eta \equiv \forall a b. DS3^{ab} \iota \eta$

**definition**  $DS4 \iota \eta \equiv \forall a b. DS4^{ab} \iota \eta$

**declare**  $DS1-def[Defs]$   $DS2-def[Defs]$   $DS3-def[Defs]$   $DS4-def[Defs]$

All DS variants are in general independent from each other. However if we take classical implication we can verify that the pairs DS1-DS3 and DS2-DS4 are indeed equivalent.

**lemma**  $DS1(\rightarrow) \eta = DS3(\rightarrow) \eta$  **unfolding** *Defs by (metis impl-def join-def)*

**lemma**  $DS2(\rightarrow) \eta = DS4(\rightarrow) \eta$  **unfolding** *Defs by (metis impl-def join-def)*

Explore some (non)entailment relations.

**lemma** *DS1-nDNor*:  $DS1(\rightarrow) \eta \implies nDNor \eta$  **unfolding** *Defs* **by** (*metis bottom-def impl-def join-def top-def*)  
**lemma** *DS2-nNor*:  $DS2(\rightarrow) \eta \implies nNor \eta$  **unfolding** *Defs* **by** (*metis bottom-def impl-def join-def top-def*)  
**lemma** *lCoP2-DS1*:  $lCoP2(\rightarrow) \eta \implies DS1(\rightarrow) \eta$  **unfolding** *Defs* **conn** **by** *metis*  
**lemma** *lCoP1-DS2*:  $lCoP1(\rightarrow) \eta \implies DS2(\rightarrow) \eta$  **unfolding** *Defs* **by** (*metis (no-types, lifting) conn*)  
**lemma** *CoP2*  $\eta \implies DS1(\rightarrow) \eta$  **nitpick oops**  
**lemma** *CoP1*  $\eta \implies DS2(\rightarrow) \eta$  **nitpick oops**

**end**

**theory** *sse-operation-negative-quantification*

**imports** *sse-operation-negative sse-boolean-algebra-quantification*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

### 3.2 Definitions and interrelations (infinitary case)

We define and interrelate infinitary variants for some previously introduced ('negative') conditions on operations. We show how they relate to quantifiers as previously defined.

iDM: infinitary De Morgan laws.

**abbreviation** *riDM1* (*iDM1<sup>-</sup>* -) **where**  $iDM1^S \eta \equiv \eta(\bigvee S) \preceq \bigwedge Ra[\eta|S]$

**abbreviation** *riDM2* (*iDM2<sup>-</sup>* -) **where**  $iDM2^S \eta \equiv \bigvee Ra[\eta|S] \preceq \eta(\bigwedge S)$

**abbreviation** *riDM3* (*iDM3<sup>-</sup>* -) **where**  $iDM3^S \eta \equiv \eta(\bigwedge S) \preceq \bigvee Ra[\eta|S]$

**abbreviation** *riDM4* (*iDM4<sup>-</sup>* -) **where**  $iDM4^S \eta \equiv \bigwedge Ra[\eta|S] \preceq \eta(\bigvee S)$

**definition** *iDM1*  $\eta \equiv \forall S. iDM1^S \eta$

**definition** *iDM2*  $\eta \equiv \forall S. iDM2^S \eta$

**definition** *iDM3*  $\eta \equiv \forall S. iDM3^S \eta$

**definition** *iDM4*  $\eta \equiv \forall S. iDM4^S \eta$

**declare** *iDM1-def*[*Defs*] *iDM2-def*[*Defs*] *iDM3-def*[*Defs*] *iDM4-def*[*Defs*]

**lemma** *CoPw-iDM1*:  $CoPw \eta \implies iDM1 \eta$  **unfolding** *Defs* **by** (*smt Ra-restr-all sup-char*)

**lemma** *CoPw-iDM2*:  $CoPw \eta \implies iDM2 \eta$  **unfolding** *Defs* **by** (*smt Ra-restr-ex inf-char*)

**lemma** *CoP2-iDM3*:  $CoP2 \eta \implies iDM3 \eta$  **by** (*smt CoP2-def Ra-restr-ex iDM3-def inf-char*)

**lemma** *CoP1-iDM4*:  $CoP1 \eta \implies iDM4 \eta$  **by** (*smt CoP1-def Ra-restr-all iDM4-def sup-char*)

**lemma** *XCoP*  $\eta \implies iDM3 \eta$  **nitpick oops**

**lemma** *XCoP*  $\eta \implies iDM4 \eta$  **nitpick oops**

DM1, DM2, iDM1, iDM2 and CoPw are equivalent.

**lemma** *iDM1-rel*:  $iDM1 \eta \implies DM1 \eta$  **proof** -

**assume** *idm1*:  $iDM1 \eta$

{ **fix** *a::σ* **and** *b::σ*

**let**  $?S = \lambda Z. Z = a \vee Z = b$

**have**  $\bigwedge Ra[\eta|?S] = \forall^R(\lambda Z. Z = a \vee Z = b) \eta$  **using** *Ra-restr-all* **by** *blast*

**moreover have**  $\forall^R(\lambda Z. Z = a \vee Z = b) \eta = (\eta a) \wedge (\eta b)$  **using** *meet-def* **by** *auto*

**ultimately have**  $\bigwedge Ra[\eta|?S] = (\eta a) \wedge (\eta b)$  **by** *simp*

**moreover have**  $\bigvee ?S = a \vee b$  **using** *supremum-def join-def* **by** *auto*

**moreover from** *idm1* **have**  $\eta(\bigvee ?S) \preceq \bigwedge Ra[\eta|?S]$  **by** (*simp add: iDM1-def*)

**ultimately have**  $\eta(a \vee b) \preceq (\eta a) \wedge (\eta b)$  **by** *simp*

} **thus** *thesis* **by** (*simp add: DM1-def*)

**qed**

**lemma** *iDM2-rel*:  $iDM2 \eta \implies DM2 \eta$  **proof** -

**assume** *idm2*:  $iDM2 \eta$

{ **fix** *a::σ* **and** *b::σ*

**let**  $?S = \lambda Z. Z = a \vee Z = b$

**have**  $\bigvee Ra[\eta|?S] = \exists^R(\{?S\}) \eta$  **using** *Ra-restr-ex* **by** *blast*  
**moreover have**  $\exists^R(\{?S\}) \eta = (\eta a) \vee (\eta b)$  **using** *join-def* **by** *auto*  
**ultimately have**  $\bigvee Ra[\eta|?S] = (\eta a) \vee (\eta b)$  **by** *simp*  
**moreover have**  $\bigwedge ?S = a \wedge b$  **using** *infimum-def meet-def* **by** *auto*  
**moreover from** *idm2* **have**  $\bigvee Ra[\eta|?S] \preceq \eta(\bigwedge ?S)$  **by** (*simp add: iDM2-def*)  
**ultimately have**  $(\eta a) \vee (\eta b) \preceq \eta(a \wedge b)$  **by** *auto*  
**}** **thus** *?thesis* **by** (*simp add: DM2-def*)  
**qed**  
**lemma** *DM1*  $\eta = iDM1 \eta$  **using** *CoPw-iDM1 DM1-CoPw iDM1-rel* **by** *blast*  
**lemma** *DM2*  $\eta = iDM2 \eta$  **using** *CoPw-iDM2 DM2-CoPw iDM2-rel* **by** *blast*  
**lemma** *iDM1*  $\eta = iDM2 \eta$  **using** *CoPw-iDM1 CoPw-iDM2 DM1-CoPw DM2-CoPw iDM1-rel iDM2-rel*  
**by** *blast*

iDM3/4 entail their finitary variants but not the other way round.

**lemma** *iDM3-rel: iDM3*  $\eta \implies DM3 \eta$  **proof** –  
**assume** *idm3: iDM3*  $\eta$   
**{ fix** *a::σ* **and** *b::σ*  
**let**  $?S = \lambda Z. Z = a \vee Z = b$   
**have**  $\bigvee Ra[\eta|?S] = \exists^R(\{?S\}) \eta$  **using** *Ra-restr-ex* **by** *blast*  
**moreover have**  $\exists^R(\{?S\}) \eta = (\eta a) \vee (\eta b)$  **using** *join-def* **by** *auto*  
**ultimately have**  $\bigvee Ra[\eta|?S] = (\eta a) \vee (\eta b)$  **by** *simp*  
**moreover have**  $\bigwedge ?S = a \wedge b$  **using** *infimum-def meet-def* **by** *auto*  
**moreover from** *idm3* **have**  $\eta(\bigwedge ?S) \preceq \bigvee Ra[\eta|?S]$  **by** (*simp add: iDM3-def*)  
**ultimately have**  $\eta(a \wedge b) \preceq (\eta a) \vee (\eta b)$  **by** *auto*  
**}** **thus** *?thesis* **by** (*simp add: DM3-def*)  
**qed**

**lemma** *iDM4-rel: iDM4*  $\eta \implies DM4 \eta$  **proof** –  
**assume** *idm4: iDM4*  $\eta$   
**{ fix** *a::σ* **and** *b::σ*  
**let**  $?S = \lambda Z. Z = a \vee Z = b$   
**have**  $\bigwedge Ra[\eta|?S] = \forall^R(\{?S\}) \eta$  **using** *Ra-restr-all* **by** *blast*  
**moreover have**  $\forall^R(\{?S\}) \eta = (\eta a) \wedge (\eta b)$  **using** *meet-def* **by** *auto*  
**ultimately have**  $\bigwedge Ra[\eta|?S] = (\eta a) \wedge (\eta b)$  **by** *simp*  
**moreover have**  $\bigvee ?S = a \vee b$  **using** *supremum-def join-def* **by** *auto*  
**moreover from** *idm4* **have**  $\bigwedge Ra[\eta|?S] \preceq \eta(\bigvee ?S)$  **by** (*simp add: iDM4-def*)  
**ultimately have**  $(\eta a) \wedge (\eta b) \preceq \eta(a \vee b)$  **by** *simp*  
**}** **thus** *?thesis* **by** (*simp add: DM4-def*)  
**qed**

**lemma** *DM3*  $\eta \implies iDM3 \eta$  **nitpick** *oops*  
**lemma** *DM4*  $\eta \implies iDM4 \eta$  **nitpick** *oops*

Indeed the previous characterization of the infinitary De Morgan laws is fairly general and entails the traditional version employing quantifiers (though not the other way round).

The first two variants DM1/2 follow easily from DM1/2, iDM1/2 or CoPw (all of them equivalent).

**lemma** *iDM1-trad: iDM1*  $\eta \implies \forall \pi. \eta(\exists x. \pi x) \preceq (\forall x. \eta(\pi x))$  **by** (*metis (mono-tags, lifting) CoPw-def DM1-CoPw iDM1-rel*)

**lemma** *iDM2-trad: iDM2*  $\eta \implies \forall \pi. (\exists x. \eta(\pi x)) \preceq \eta(\forall x. \pi x)$  **by** (*metis (mono-tags, lifting) CoPw-def DM2-CoPw iDM2-rel*)

An analogous relationship holds for variants DM3/4, though the proof is less trivial. To see how let us first consider an intermediate version of the De Morgan laws, obtained as a particular case of the general variant above, with S as the range of a propositional function.

**abbreviation** *piDM1*  $\pi \eta \equiv \eta(\bigvee Ra(\pi)) \preceq \bigwedge Ra[\eta|Ra(\pi)]$

**abbreviation**  $piDM2 \ \pi \ \eta \equiv \bigvee Ra[\eta|Ra(\pi)] \preceq \eta(\bigwedge Ra(\pi))$   
**abbreviation**  $piDM3 \ \pi \ \eta \equiv \eta(\bigwedge Ra(\pi)) \preceq \bigvee Ra[\eta|Ra(\pi)]$   
**abbreviation**  $piDM4 \ \pi \ \eta \equiv \bigwedge Ra[\eta|Ra(\pi)] \preceq \eta(\bigvee Ra(\pi))$

They are entailed (unidirectionally) by the general De Morgan laws.

**lemma**  $iDM1 \ \eta \implies \forall \pi. \ piDM1 \ \pi \ \eta$  **by** (*simp add: iDM1-def*)  
**lemma**  $iDM2 \ \eta \implies \forall \pi. \ piDM2 \ \pi \ \eta$  **by** (*simp add: iDM2-def*)  
**lemma**  $iDM3 \ \eta \implies \forall \pi. \ piDM3 \ \pi \ \eta$  **by** (*simp add: iDM3-def*)  
**lemma**  $iDM4 \ \eta \implies \forall \pi. \ piDM4 \ \pi \ \eta$  **by** (*simp add: iDM4-def*)

Drawing upon the relationships shown previously we can rewrite the latter two as:

**lemma**  $iDM3\text{-aux}: \ piDM3 \ \pi \ \eta \equiv \eta(\forall \pi) \preceq \exists [\eta|(\bigwedge Ra \ \pi)]^N$  **unfolding** *Ra-all Ra-ex-restr* **by** *simp*  
**lemma**  $iDM4\text{-aux}: \ piDM4 \ \pi \ \eta \equiv \forall [\eta|(\bigvee Ra \ \pi)]^P \preceq \eta(\exists \pi)$  **unfolding** *Ra-ex Ra-all-restr* **by** *simp*

and thus finally obtain the desired formulas.

**lemma**  $iDM3\text{-trad}: \ iDM3 \ \eta \implies \forall \pi. \ \eta(\forall x. \ \pi \ x) \preceq (\exists x. \ \eta(\pi \ x))$  **by** (*metis Ra-ex-comp2 iDM3-def iDM3-aux comp-apply*)  
**lemma**  $iDM4\text{-trad}: \ iDM4 \ \eta \implies \forall \pi. \ (\forall x. \ \eta(\pi \ x)) \preceq \eta(\exists x. \ \pi \ x)$  **by** (*metis Ra-all-comp1 iDM4-def iDM4-aux comp-apply*)

**end**

**theory** *topo-operators-basic*

**imports** *sse-operation-positive-quantification*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

**abbreviation**  $implies\text{-rl}::\text{bool} \implies \text{bool} \implies \text{bool}$  (**infixl**  $\longleftarrow$  25) **where**  $\varphi \longleftarrow \psi \equiv \psi \longrightarrow \varphi$

## 4 Topological operators

Below we define some conditions on algebraic operations (aka. operators) with type  $\sigma \implies \sigma$ . Those operations are aimed at extending a Boolean 'algebra of propositions' towards different generalizations of topological algebras. We divide this section into two parts. In the first we define and interrelate the topological operators of interior, closure, border and frontier. In the second we introduce the (more fundamental) notion of derivative (aka. derived set) and its related notion of (Cantorian) coherence, defining both as operators. We follow the naming conventions introduced originally by Kuratowski [8] (cf. also [9]) and Zarycki [12].

### 4.1 Interior and closure

In this section we examine the traditional notion of topological (closure, resp. interior) algebras in the spirit of McKinsey & Tarski [11], but drawing primarily from the works of Zarycki [12] and Kuratowski [8]. We also explore the less-known notions of border (cf. 'Rand' [6], 'bord' [12]) and frontier (aka. 'boundary'; cf. 'Grenze' [6], 'frontière' [12] [9]) as studied by Zarycki [12] and define corresponding operations for them.

#### 4.1.1 Interior conditions

**abbreviation**  $Int\text{-1} \ \varphi \equiv MULT \ \varphi$   
**abbreviation**  $Int\text{-1a} \ \varphi \equiv MULT\text{-a} \ \varphi$   
**abbreviation**  $Int\text{-1b} \ \varphi \equiv MULT\text{-b} \ \varphi$   
**abbreviation**  $Int\text{-2} \ \varphi \equiv dEXP \ \varphi$

**abbreviation** *Int-3*  $\varphi \equiv dNOR \varphi$   
**abbreviation** *Int-4*  $\varphi \equiv IDEM \varphi$   
**abbreviation** *Int-4'*  $\varphi \equiv IDEMa \varphi$

**abbreviation** *Int-5*  $\varphi \equiv NOR \varphi$   
**definition** *Int-6*  $\varphi \equiv \forall A B. \varphi(A \leftarrow B) \preceq \varphi(A) \leftarrow \varphi(B)$   
**definition** *Int-7*  $\varphi \equiv \forall A B. \varphi(A \rightarrow B) \preceq \varphi(A) \rightarrow \varphi(B)$   
**definition** *Int-8*  $\varphi \equiv \forall A B. \varphi(\varphi A \vee \varphi B) \approx (\varphi A) \vee (\varphi B)$   
**definition** *Int-9*  $\varphi \equiv \forall A B. \varphi A \preceq B \longrightarrow \varphi A \preceq \varphi B$

$\varphi$  is an interior operator ( $\mathfrak{J}(\varphi)$ ) iff it satisfies conditions 1-4 (cf. [12] and also [9]). This characterization is shown consistent by generating a non-trivial model.

**abbreviation**  $\mathfrak{J} \varphi \equiv Int-1 \varphi \wedge Int-2 \varphi \wedge Int-3 \varphi \wedge Int-4 \varphi$   
**lemma**  $\mathfrak{J} \varphi$  **nitpick**[*satisfy, card w=3*] **oops**

We verify some properties which will become useful later (also to improve provers' performance).

**lemma** *PI1*: *Int-1*  $\varphi = (Int-1a \varphi \wedge Int-1b \varphi)$  **by** (*metis MULT-def MULT-a-def MULT-b-def*)  
**lemma** *PI4*: *Int-2*  $\varphi \implies (Int-4 \varphi = Int-4' \varphi)$  **unfolding** *dEXP-def IDEM-def IDEMa-def* **by** *blast*  
**lemma** *PI5*: *Int-2*  $\varphi \implies Int-5 \varphi$  **unfolding** *dEXP-def NOR-def conn* **by** *blast*  
**lemma** *PI6*: *Int-1a*  $\varphi \implies Int-2 \varphi \implies Int-6 \varphi$  **by** (*smt dEXP-def Int-6-def MONO-MULTa MONO-def conn*)  
**lemma** *PI7*: *Int-1*  $\varphi \implies Int-7 \varphi$  **proof** –  
  **assume** *int1*: *Int-1*  $\varphi$   
  { **fix** *a b*  
    **have**  $a \wedge b = a \wedge (a \rightarrow b)$  **unfolding** *conn* **by** *blast*  
    **hence**  $\varphi(a \wedge b) = \varphi(a \wedge (a \rightarrow b))$  **by** *simp*  
    **moreover from** *int1* **have**  $\varphi(a \wedge b) \approx \varphi a \wedge \varphi b$  **by** (*simp add: MULT-def*)  
    **moreover from** *int1* **have**  $\varphi(a \wedge (a \rightarrow b)) \approx \varphi a \wedge \varphi(a \rightarrow b)$  **by** (*simp add: MULT-def*)  
    **ultimately have**  $\varphi a \wedge \varphi(a \rightarrow b) \approx \varphi a \wedge \varphi b$  **by** *simp*  
    **hence**  $\varphi a \wedge \varphi(a \rightarrow b) \approx \varphi a \wedge (\varphi a \rightarrow \varphi b)$  **unfolding** *conn* **by** *blast*  
    **hence**  $\varphi(a \rightarrow b) \preceq \varphi a \rightarrow \varphi b$  **unfolding** *conn* **by** *blast*  
  } **thus** *?thesis* **by** (*simp add: Int-7-def*)  
**qed**  
**lemma** *PI8*: *Int-1a*  $\varphi \implies Int-2 \varphi \implies Int-4 \varphi \implies Int-8 \varphi$  **using** *ADDI-b-def IDEM-def Int-8-def MONO-ADDIb MONO-MULTa dEXP-def join-def* **by** *auto*  
**lemma** *PI9*: *Int-1a*  $\varphi \implies Int-4 \varphi \implies Int-9 \varphi$  **using** *IDEM-def Int-9-def MONO-def MONO-MULTa* **by** *simp*

#### 4.1.2 Closure conditions

**abbreviation** *Cl-1*  $\varphi \equiv ADDI \varphi$   
**abbreviation** *Cl-1a*  $\varphi \equiv ADDI-a \varphi$   
**abbreviation** *Cl-1b*  $\varphi \equiv ADDI-b \varphi$   
**abbreviation** *Cl-2*  $\varphi \equiv EXP \varphi$   
**abbreviation** *Cl-3*  $\varphi \equiv NOR \varphi$   
**abbreviation** *Cl-4*  $\varphi \equiv IDEM \varphi$   
**abbreviation** *Cl-4'*  $\varphi \equiv IDEMb \varphi$

**abbreviation** *Cl-5*  $\varphi \equiv dNOR \varphi$   
**definition** *Cl-6*  $\varphi \equiv \forall A B. (\varphi A) \leftarrow (\varphi B) \preceq \varphi(A \leftarrow B)$   
**definition** *Cl-7*  $\varphi \equiv \forall A B. (\varphi A) \rightarrow (\varphi B) \preceq \varphi(A \rightarrow B)$   
**definition** *Cl-8*  $\varphi \equiv \forall A B. \varphi(\varphi A \wedge \varphi B) \approx (\varphi A) \wedge (\varphi B)$   
**definition** *Cl-9*  $\varphi \equiv \forall A B. A \preceq \varphi B \longrightarrow \varphi A \preceq \varphi B$

$\varphi$  is a closure operator ( $\mathfrak{C}(\varphi)$ ) iff it satisfies conditions 1-4 (cf. [8] [9]). This characterization is shown consistent by generating a non-trivial model.



abbreviation  $\mathfrak{C} \varphi \equiv Cl-1 \varphi \wedge Cl-2 \varphi \wedge Cl-3 \varphi \wedge Cl-4 \varphi$

lemma  $\mathfrak{C} \varphi$  nitpick[satisfy, card w=3] oops

We verify some properties that will become useful later.

lemma PC1:  $Cl-1 \varphi = (Cl-1a \varphi \wedge Cl-1b \varphi)$  unfolding ADDI-def ADDI-a-def ADDI-b-def by blast

lemma PC4:  $Cl-2 \varphi \implies (Cl-4 \varphi = Cl-4' \varphi)$  unfolding EXP-def IDEM-def IDEMb-def by smt

lemma PC5:  $Cl-2 \varphi \implies Cl-5 \varphi$  unfolding EXP-def dNOR-def conn by simp

lemma PC6:  $Cl-1 \varphi \implies Cl-6 \varphi$  proof –

assume cl1:  $Cl-1 \varphi$

{ fix a b

have  $a \vee b = (a \leftarrow b) \vee b$  unfolding conn by blast

hence  $\varphi(a \vee b) = \varphi((a \leftarrow b) \vee b)$  by simp

moreover from cl1 have  $\varphi(a \vee b) \approx \varphi a \vee \varphi b$  by (simp add: ADDI-def)

moreover from cl1 have  $\varphi((a \leftarrow b) \vee b) \approx \varphi(a \leftarrow b) \vee (\varphi b)$  by (simp add: ADDI-def)

ultimately have  $\varphi a \vee \varphi b \approx \varphi(a \leftarrow b) \vee \varphi b$  by simp

hence  $(\varphi a \leftarrow \varphi b) \vee \varphi b \approx \varphi(a \leftarrow b) \vee \varphi b$  unfolding conn by blast

hence  $\varphi a \leftarrow \varphi b \preceq \varphi(a \leftarrow b)$  unfolding conn by blast

} thus ?thesis by (simp add: Cl-6-def)

qed

lemma PC7:  $Cl-1b \varphi \implies Cl-2 \varphi \implies Cl-7 \varphi$  by (smt EXP-def Cl-7-def MONO-def PC1 MONO-ADDIb conn)

lemma PC8:  $Cl-1b \varphi \implies Cl-2 \varphi \implies Cl-4 \varphi \implies Cl-8 \varphi$  by (smt Cl-8-def EXP-def IDEM-def MONO-ADDIb MONO-MULTa MULT-a-def meet-def)

lemma PC9:  $Cl-1b \varphi \implies Cl-4 \varphi \implies Cl-9 \varphi$  by (smt IDEM-def Cl-9-def MONO-def MONO-ADDIb)

### 4.1.3 Exploring dualities

lemma IC1-dual:  $Int-1a \varphi = Cl-1b \varphi$  using MONO-ADDIb MONO-MULTa by auto

lemma Int-1b  $\varphi = Cl-1a \varphi$  nitpick oops

lemma IC1a:  $Int-1a \varphi \implies Cl-1b \varphi^d$  by (smt MULT-a-def ADDI-b-def MONO-def MONO-MULTa dual-def conn)

lemma IC1b:  $Int-1b \varphi \implies Cl-1a \varphi^d$  unfolding MULT-b-def ADDI-a-def dual-def conn by auto

lemma IC1:  $Int-1 \varphi \implies Cl-1 \varphi^d$  unfolding MULT-def ADDI-def dual-def conn by simp

lemma IC2:  $Int-2 \varphi \implies Cl-2 \varphi^d$  unfolding dEXP-def EXP-def dual-def conn by blast

lemma IC3:  $Int-3 \varphi \implies Cl-3 \varphi^d$  unfolding dNOR-def NOR-def dual-def conn by simp

lemma IC4:  $Int-4 \varphi \implies Cl-4 \varphi^d$  unfolding IDEM-def IDEM-def dual-def conn by simp

lemma IC4':  $Int-4' \varphi \implies Cl-4' \varphi^d$  unfolding IDEMa-def IDEMb-def dual-def conn by metis

lemma IC5:  $Int-5 \varphi \implies Cl-5 \varphi^d$  unfolding NOR-def dNOR-def dual-def conn by simp

lemma CI1a:  $Cl-1a \varphi \implies Int-1b \varphi^d$  proof –

assume cl1a:  $Cl-1a \varphi$

{ fix A B

have  $-\varphi(-(A \wedge B)) \approx -\varphi(-A \vee -B)$  unfolding conn by simp

moreover from cl1a have  $-(\varphi(-A) \vee \varphi(-B)) \preceq -\varphi(-A \vee -B)$  using ADDI-a-def conn by

metis

ultimately have  $-(\varphi(-A)) \wedge -(\varphi(-B)) \preceq -\varphi(-(A \wedge B))$  unfolding conn by simp

hence  $(\varphi^d A) \wedge (\varphi^d B) \preceq (\varphi^d (A \wedge B))$  unfolding dual-def by simp

} thus Int-1b  $\varphi^d$  using MULT-b-def by blast

qed

lemma CI1b:  $Cl-1b \varphi \implies Int-1a \varphi^d$  by (metis IC1a MONO-ADDIb MONO-MULTa)

lemma CI1:  $Cl-1 \varphi \implies Int-1 \varphi^d$  by (simp add: CI1a CI1b PC1 PI1)

lemma CI2:  $Cl-2 \varphi \implies Int-2 \varphi^d$  unfolding EXP-def dEXP-def dual-def conn by metis

lemma CI3:  $Cl-3 \varphi \implies Int-3 \varphi^d$  unfolding NOR-def dNOR-def dual-def conn by simp

lemma CI4:  $Cl-4 \varphi \implies Int-4 \varphi^d$  unfolding IDEM-def IDEM-def dual-def conn by simp

lemma CI4':  $Cl-4' \varphi \implies Int-4' \varphi^d$  unfolding IDEMa-def IDEMb-def dual-def conn by metis

lemma *CI5*:  $Cl\text{-}5 \varphi \implies Int\text{-}5 \varphi^d$  **unfolding** *dNOR-def NOR-def dual-def conn by simp*

## 4.2 Frontier and border

### 4.2.1 Frontier conditions

**definition** *Fr-1a*  $\varphi \equiv \forall A B. (A \wedge B) \wedge \varphi(A \wedge B) \preceq (A \wedge B) \wedge (\varphi A \vee \varphi B)$

**definition** *Fr-1b*  $\varphi \equiv \forall A B. (A \wedge B) \wedge \varphi(A \wedge B) \succeq (A \wedge B) \wedge (\varphi A \vee \varphi B)$

**definition** *Fr-1*  $\varphi \equiv \forall A B. (A \wedge B) \wedge \varphi(A \wedge B) \approx (A \wedge B) \wedge (\varphi A \vee \varphi B)$

**definition** *Fr-2*  $\varphi \equiv \forall A. \varphi A \approx \varphi(\neg A)$

**abbreviation** *Fr-3*  $\varphi \equiv NOR \varphi$

**definition** *Fr-4*  $\varphi \equiv \forall A. \varphi(\varphi A) \preceq (\varphi A)$

**definition** *Fr-5*  $\varphi \equiv \forall A. \varphi(\varphi(\varphi A)) \approx \varphi(\varphi A)$

**definition** *Fr-6*  $\varphi \equiv \forall A B. A \preceq B \longrightarrow (\varphi A \preceq B \vee \varphi B)$

$\varphi$  is a topological frontier operator ( $\mathfrak{F}(\varphi)$ ) iff it satisfies conditions 1-4 (cf. [12]). This is also shown consistent by generating a non-trivial model.

**abbreviation**  $\mathfrak{F} \varphi \equiv Fr\text{-}1 \varphi \wedge Fr\text{-}2 \varphi \wedge Fr\text{-}3 \varphi \wedge Fr\text{-}4 \varphi$

**lemma**  $\mathfrak{F} \varphi$  **nitpick**[*satisfy, card w=3*] **oops**

We now verify some useful properties of the frontier operator.

**lemma** *PF1*:  $Fr\text{-}1 \varphi = (Fr\text{-}1a \varphi \wedge Fr\text{-}1b \varphi)$  **unfolding** *Fr-1-def Fr-1a-def Fr-1b-def by meson*

**lemma** *PF5*:  $Fr\text{-}1 \varphi \implies Fr\text{-}4 \varphi \implies Fr\text{-}5 \varphi$  **proof** –

assume *fr1*:  $Fr\text{-}1 \varphi$  and *fr4*:  $Fr\text{-}4 \varphi$

{ fix  $A$

from *fr1* have  $(\varphi(\varphi A) \wedge (\varphi A)) \wedge \varphi(\varphi(\varphi A) \wedge (\varphi A)) \approx (\varphi(\varphi A) \wedge (\varphi A)) \wedge (\varphi(\varphi(\varphi A)) \vee \varphi(\varphi A))$  by (*simp add: Fr-1-def*)

moreover have *r1*:  $\varphi(\varphi A) \wedge (\varphi A) \approx \varphi(\varphi A)$  **using** *meet-char Fr-4-def fr4 by simp*

moreover have *r2*:  $\varphi(\varphi(\varphi A)) \vee \varphi(\varphi A) \approx \varphi(\varphi A)$  **using** *join-char Fr-4-def fr4 by simp*

ultimately have  $(\varphi(\varphi A) \wedge (\varphi A)) \wedge \varphi(\varphi(\varphi A)) \approx (\varphi(\varphi A) \wedge (\varphi A)) \wedge \varphi(\varphi A)$  **unfolding conn**

by *auto*

hence  $\varphi(\varphi(\varphi A)) \approx \varphi(\varphi A)$  **using** *r1 r2 conn by auto*

} thus *?thesis* by (*simp add: Fr-5-def*)

qed

**lemma** *PF6*:  $Fr\text{-}1b \varphi \implies Fr\text{-}2 \varphi \implies Fr\text{-}6 \varphi$  **proof** –

assume *fr1b*:  $Fr\text{-}1b \varphi$  and *fr2*:  $Fr\text{-}2 \varphi$

{ fix  $A B$

have  $\varphi(\neg(A \vee B)) \approx \varphi(\neg A \wedge \neg B)$  **unfolding conn by simp**

hence *aux*:  $\varphi(\neg A \wedge \neg B) \approx \varphi(A \vee B)$  **by** (*metis (mono-tags) Fr-2-def fr2*)

from *fr1b* have  $(\neg A \wedge \neg B) \wedge \varphi(\neg A \wedge \neg B) \succeq (\neg A \wedge \neg B) \wedge (\varphi(\neg A) \vee \varphi(\neg B))$  **using** *Fr-1b-def by metis*

hence  $A \vee B \vee \neg\varphi(\neg A \wedge \neg B) \preceq A \vee B \vee (\neg(\varphi A) \wedge \neg(\varphi B))$  **using** *Fr-2-def fr2 conn by auto*

hence  $\neg A \wedge \neg B \wedge \neg\varphi(\neg A \wedge \neg B) \preceq \neg A \wedge \neg B \wedge \neg(\varphi A) \wedge \neg(\varphi B)$  **unfolding conn by blast**

hence  $A \vee B \vee \varphi(A \vee B) \succeq A \vee B \vee (\varphi A) \vee (\varphi B)$  **using** *aux unfolding conn by blast*

hence  $A \preceq B \longrightarrow B \vee \varphi(A \vee B) \succeq B \vee (\varphi A) \vee (\varphi B)$  **unfolding conn by auto**

hence  $A \preceq B \longrightarrow B \vee (\varphi B) \succeq B \vee (\varphi A) \vee (\varphi B)$  **using** *join-char unfolding conn by simp*

hence  $A \preceq B \longrightarrow (\varphi A) \preceq B \vee (\varphi B)$  **unfolding conn by simp**

} thus  $Fr\text{-}6 \varphi$  by (*simp add: Fr-6-def*)

qed

### 4.2.2 Border conditions

**definition** *Br-1*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \approx (A \wedge \varphi B) \vee (B \wedge \varphi A)$

**definition** *Br-2*  $\varphi \equiv (\varphi \top) \approx \perp$

**definition** *Br-3*  $\varphi \equiv \forall A. \varphi(\varphi^d A) \preceq A$

**definition** *Br-4*  $\varphi \equiv \forall A B. A \preceq B \longrightarrow A \wedge (\varphi B) \preceq \varphi A$

**definition** *Br-5a*  $\varphi \equiv \forall A. \varphi(\varphi^d A) \preceq \varphi A$

**definition** *Br-5b*  $\varphi \equiv \forall A. \varphi A \preceq A$

**definition** *Br-5c*  $\varphi \equiv \forall A. A \preceq \varphi^d A$

**definition** *Br-5d*  $\varphi \equiv \forall A. \varphi^d A \preceq \varphi^d(\varphi A)$

**abbreviation** *Br-6*  $\varphi \equiv \text{IDEM } \varphi$

**abbreviation** *Br-7*  $\varphi \equiv \text{ADDI-a } \varphi$

**abbreviation** *Br-8*  $\varphi \equiv \text{MULT-b } \varphi$

**definition** *Br-9*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \preceq (\varphi A) \vee (\varphi B)$

**definition** *Br-10*  $\varphi \equiv \forall A. \varphi(\neg(\varphi A) \wedge \varphi^d A) \approx \perp$

$\varphi$  is a topological border operator ( $\mathfrak{B}(\varphi)$ ) iff it satisfies conditions 1-3 (cf. [12]). This is also shown consistent.

**abbreviation**  $\mathfrak{B} \varphi \equiv \text{Br-1 } \varphi \wedge \text{Br-2 } \varphi \wedge \text{Br-3 } \varphi$

**lemma**  $\mathfrak{B} \varphi$  **nitpick**[*satisfy, card w=3*] **oops**

We now verify some useful properties of the border operator.

**lemma** *PB4*: *Br-1*  $\varphi \implies \text{Br-4 } \varphi$  **proof** –

assume *br1*: *Br-1*  $\varphi$

{ fix *A B*

have *aux*:  $\varphi(A \wedge B) \approx (A \wedge \varphi B) \vee (B \wedge \varphi A)$  **using** *br1 Br-1-def* **by** *simp*

{ assume  $A \preceq B$

hence  $\varphi(A \wedge B) \approx \varphi A$  **using** *meet-char unfolding conn* **by** *simp*

hence  $\varphi A \approx (A \wedge \varphi B) \vee (B \wedge \varphi A)$  **using** *aux* **by** *auto*

hence  $A \wedge \varphi B \preceq \varphi A$  **unfolding conn** **by** *auto*

} hence  $A \preceq B \longrightarrow A \wedge \varphi B \preceq \varphi A$  **by** (*rule impI*)

} thus *Br-4*  $\varphi$  **by** (*simp add: Br-4-def*)

qed

**lemma** *PB5b*: *Br-1*  $\varphi \implies \text{Br-5b } \varphi$  **proof** –

assume *br1*: *Br-1*  $\varphi$

{ fix *A*

from *br1* have *aux*:  $\varphi(A \wedge A) \approx (A \wedge \varphi A) \vee (A \wedge \varphi A)$  **unfolding** *Br-1-def* **by** *blast*

hence  $\varphi A \approx (A \wedge \varphi A)$  **unfolding conn** **by** *metis*

hence  $\varphi A \preceq A$  **unfolding conn** **by** *blast*

} thus *Br-5b*  $\varphi$  **using** *Br-5b-def* **by** *blast*

qed

**lemma** *PB5c*: *Br-1*  $\varphi \implies \text{Br-5c } \varphi$  **using** *Br-5b-def Br-5c-def PB5b dual-def* **unfolding conn** **by** *force*

**lemma** *PB5a*: *Br-1*  $\varphi \implies \text{Br-3 } \varphi \implies \text{Br-5a } \varphi$  **proof** –

assume *br1*: *Br-1*  $\varphi$  **and** *br3*: *Br-3*  $\varphi$

hence *br5c*: *Br-5c*  $\varphi$  **using** *PB5c* **by** *simp*

{ fix *A*

have  $A \wedge \varphi(\varphi^d A) \preceq \varphi A$  **by** (*metis br5c Br-4-def Br-5c-def PB4 br1*)

hence  $\varphi(\varphi^d A) \preceq \varphi A$  **using** *Br-3-def br3* **unfolding conn** **by** *metis*

} thus *Br-5a*  $\varphi$  **using** *Br-5a-def* **by** *simp*

qed

**lemma** *PB5d*: *Br-1*  $\varphi \implies \text{Br-3 } \varphi \implies \text{Br-5d } \varphi$  **proof** –

assume *br1*: *Br-1*  $\varphi$  **and** *br3*: *Br-3*  $\varphi$

hence *br5a*: *Br-5a*  $\varphi$  **using** *PB5a* **by** *simp*

{ fix *A*

from *br5a* have  $\varphi(\varphi^d(-A)) \preceq \varphi(-A)$  **unfolding** *Br-5a-def* **by** *simp*

hence  $\neg\varphi(-A) \preceq \neg\varphi(\varphi^d(-A))$  **unfolding conn** **by** *blast*

hence  $\varphi^d A \preceq \varphi^d(\varphi A)$  **unfolding dual-def conn** **by** *simp*

} thus *Br-5d*  $\varphi$  **using** *Br-5d-def* **by** *simp*

qed

**lemma** *PB6*:  $Br-1 \varphi \implies Br-6 \varphi$  **by** (*smt Br-4-def Br-5b-def IDEM-def PB4 PB5b conn*)

**lemma** *PB7*:  $Br-1 \varphi \implies Br-7 \varphi$  **using** *Br-4-def Br-5b-def ADDI-a-def PB4 PB5b unfolding conn* **by** *smt*

**lemma** *PB8*:  $Br-1 \varphi \implies Br-8 \varphi$  **using** *Br-1-def Br-5b-def MULT-b-def PB5b unfolding conn* **by** *metis*

**lemma** *PB9*:  $Br-1 \varphi \implies Br-9 \varphi$  **unfolding** *Br-1-def Br-9-def conn* **by** *simp*

**lemma** *PB10*:  $Br-1 \varphi \implies Br-3 \varphi \implies Br-10 \varphi$  **proof** — proof automagically generated

**assume** *a1*:  $Br-3 \varphi$

**assume** *a2*:  $Br-1 \varphi$

{ **fix** *bb* ::  $w \implies bool$  **and** *ww* ::  $w$

**have**  $\forall p \ pa \ w \ pb. ((pb \ p \ w \vee \ pb \ pa \ w) \vee \neg pb \ (pa \wedge \ p) \ w) \vee \neg Br-9 \ pb$

**by** (*metis Br-9-def join-def*)

**then have**  $(\varphi \ (((\varphi^c) \sqcap (\varphi^d)) \ bb) \ ww) \wedge (\perp \ ww) \vee \neg(\varphi \ (((\varphi^c) \sqcap (\varphi^d)) \ bb) \ ww) \wedge \neg(\perp \ ww)$

**using** *a2 a1* **by** (*metis (no-types) Br-5a-def Br-5b-def Br-5d-def PB5a PB5b PB5d PB9 bottom-def compl-def dual-def meet-def*)

} **then show** *?thesis* **unfolding** *Br-10-def* **by** *blast*

qed

### 4.2.3 Relation with closure and interior

We define and verify some conversion operators useful to derive border and frontier operators from closure/interior operators and also between each other.

Frontier operator as derived from interior.

**definition** *Fr-int*:: $(\sigma \implies \sigma) \implies (\sigma \implies \sigma)$  ( $\mathcal{F}_I$ ) **where**  $\mathcal{F}_I \ \mathcal{I} \equiv \lambda A. \neg(\mathcal{I} \ A) \wedge \mathcal{I}^d \ A$

**lemma** *FI1*:  $Int-1 \varphi \implies Int-2 \varphi \implies Fr-1(\mathcal{F}_I \ \varphi)$  **using** *EXP-def Fr-1-def Fr-int-def IC2 MULT-def conn* **by** *auto*

**lemma** *FI2*:  $Fr-2(\mathcal{F}_I \ \varphi)$  **using** *Fr-2-def Fr-int-def dual-def dual-symm* **unfolding** *conn* **by** *smt*

**lemma** *FI3*:  $Int-3 \varphi \implies Fr-3(\mathcal{F}_I \ \varphi)$  **using** *NOR-def Fr-int-def IC3* **unfolding** *conn* **by** *auto*

**lemma** *FI4*:  $Int-1a \varphi \implies Int-2 \varphi \implies Int-4 \varphi \implies Fr-4(\mathcal{F}_I \ \varphi)$  **unfolding** *Fr-int-def Fr-4-def dual-def conn* **by** (*smt Int-9-def MONO-MULTa PI9*)

Frontier operator as derived from closure.

**definition** *Fr-cl*:: $(\sigma \implies \sigma) \implies (\sigma \implies \sigma)$  ( $\mathcal{F}_C$ ) **where**  $\mathcal{F}_C \ \mathcal{C} \equiv \lambda A. (\mathcal{C} \ A) \wedge \mathcal{C}(\neg A)$

**lemma** *FC1*:  $Cl-1 \varphi \implies Cl-2 \varphi \implies Fr-1(\mathcal{F}_C \ \varphi)$  **using** *CI1 EXP-def Fr-1-def Fr-cl-def MULT-def dual-def* **unfolding** *conn* **by** *smt*

**lemma** *FC2*:  $Fr-2(\mathcal{F}_C \ \varphi)$  **using** *Fr-2-def Fr-cl-def dual-def dual-symm* **unfolding** *conn* **by** *smt*

**lemma** *FC3*:  $Cl-3 \varphi \implies Fr-3(\mathcal{F}_C \ \varphi)$  **unfolding** *NOR-def Fr-cl-def conn* **by** *simp*

**lemma** *FC4*:  $Cl-1b \varphi \implies Cl-2 \varphi \implies Cl-4 \varphi \implies Fr-4(\mathcal{F}_C \ \varphi)$  **using** *Cl-8-def MONO-ADDIb PC8* **unfolding** *Fr-cl-def Fr-4-def conn* **by** *blast*

Frontier operator as derived from border.

**definition** *Fr-br*:: $(\sigma \implies \sigma) \implies (\sigma \implies \sigma)$  ( $\mathcal{F}_B$ ) **where**  $\mathcal{F}_B \ \mathcal{B} \equiv \lambda A. \mathcal{B} \ A \vee \mathcal{B}(\neg A)$

**lemma** *FB1*:  $Br-1 \varphi \implies Fr-1(\mathcal{F}_B \ \varphi)$  **unfolding** *Fr-br-def Fr-1-def* **using** *Br-1-def Br-5b-def PB5b* **conn** **by** *smt*

**lemma** *FB2*:  $Fr-2(\mathcal{F}_B \ \varphi)$  **unfolding** *Fr-br-def Fr-2-def conn* **by** *auto*

**lemma** *FB3*:  $Br-1 \varphi \implies Br-2 \varphi \implies Fr-3(\mathcal{F}_B \ \varphi)$  **using** *Br-2-def Br-5b-def PB5b* **unfolding** *Fr-br-def NOR-def conn* **by** *force*

**lemma** *FB4*:  $Br-1 \varphi \implies Br-3 \varphi \implies Fr-4(\mathcal{F}_B \ \varphi)$  **proof** —

**assume** *br1*:  $Br-1 \varphi$  **and** *br3*:  $Br-3 \varphi$

{ **fix** *A*

**have**  $(\mathcal{F}_B \ \varphi) \ ((\mathcal{F}_B \ \varphi) \ A) = \varphi(\varphi \ A \vee \varphi(\neg A)) \vee \varphi(\neg(\varphi \ A \vee \varphi(\neg A)))$  **by** (*simp add: Fr-br-def conn*)

**also have**  $\dots = \varphi(\varphi \ A \vee \varphi(\neg A)) \vee \varphi(\neg(\varphi \ A) \wedge \varphi^d \ A)$  **by** (*simp add: dual-def conn*)

**also from** *br1 br3* **have**  $\dots = \varphi(\varphi \ A \vee \varphi(\neg A)) \vee \perp$  **using** *Br-10-def PB10* **by** *metis*

**finally have**  $(\mathcal{F}_B \varphi) ((\mathcal{F}_B \varphi) A) \approx \varphi(\varphi A \vee \varphi(-A))$  **unfolding conn by simp**  
**hence**  $(\mathcal{F}_B \varphi) ((\mathcal{F}_B \varphi) A) \preceq (\mathcal{F}_B \varphi) A$  **using** *Br-5b-def Fr-br-def PB5b br1* **by fastforce**  
**} thus**  $Fr-4(\mathcal{F}_B \varphi)$  **using** *Fr-4-def* **by blast**  
**qed**

Border operator as derived from interior.

**definition**  $Br-int::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma) (\mathcal{B}_I)$  **where**  $\mathcal{B}_I \mathcal{I} \equiv \lambda A. A \leftarrow (\mathcal{I} A)$   
**lemma** *BI1: Int-1*  $\varphi \Longrightarrow Br-1 (\mathcal{B}_I \varphi)$  **using** *Br-1-def Br-int-def MULT-def conn* **by auto**  
**lemma** *BI2: Int-3*  $\varphi \Longrightarrow Br-2 (\mathcal{B}_I \varphi)$  **by** (*simp add: Br-2-def Br-int-def dNOR-def conn*)  
**lemma** *BI3: Int-1a*  $\varphi \Longrightarrow Int-4 \varphi \Longrightarrow Br-3 (\mathcal{B}_I \varphi)$  **unfolding** *Br-int-def Br-3-def dual-def* **by** (*smt Int-9-def MONO-MULTa PI9 conn*)

Border operator as derived from closure.

**definition**  $Br-cl::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma) (\mathcal{B}_C)$  **where**  $\mathcal{B}_C \mathcal{C} \equiv \lambda A. A \wedge \mathcal{C}(-A)$   
**lemma** *BC1: Cl-1*  $\varphi \Longrightarrow Br-1(\mathcal{B}_C \varphi)$  **using** *Br-1-def Br-cl-def C11 MULT-def dual-def* **unfolding conn by smt**  
**lemma** *BC2: Cl-3*  $\varphi \Longrightarrow Br-2(\mathcal{B}_C \varphi)$  **using** *Br-2-def Br-cl-def C13 dNOR-def dual-def* **unfolding conn by auto**  
**lemma** *BC3: Cl-1b*  $\varphi \Longrightarrow Cl-4 \varphi \Longrightarrow Br-3 (\mathcal{B}_C \varphi)$  **unfolding** *Br-cl-def Br-3-def dual-def conn* **by** (*metis (mono-tags, lifting) Cl-9-def PC9*)

Note that the previous two conversion functions are related:

**lemma** *BI-BC-rel:*  $(\mathcal{B}_I \varphi) = \mathcal{B}_C(\varphi^d)$  **by** (*simp add: Br-cl-def Br-int-def dual-def conn*)

Border operator as derived from frontier.

**definition**  $Br-fr::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma) (\mathcal{B}_F)$  **where**  $\mathcal{B}_F \mathcal{F} \equiv \lambda A. A \wedge (\mathcal{F} A)$   
**lemma** *BF1: Fr-1*  $\varphi \Longrightarrow Br-1(\mathcal{B}_F \varphi)$  **using** *Br-1-def Br-fr-def Fr-1-def conn* **by auto**  
**lemma** *BF2: Fr-2*  $\varphi \Longrightarrow Fr-3 \varphi \Longrightarrow Br-2(\mathcal{B}_F \varphi)$  **using** *Br-2-def Br-fr-def Fr-2-def NOR-def* **unfolding conn by fastforce**  
**lemma** *BF3: Fr-1b*  $\varphi \Longrightarrow Fr-2 \varphi \Longrightarrow Fr-4 \varphi \Longrightarrow Br-3(\mathcal{B}_F \varphi)$  **proof** –  
**assume** *fr1b: Fr-1b*  $\varphi$  **and** *fr2: Fr-2*  $\varphi$  **and** *fr4: Fr-4*  $\varphi$   
**{ fix**  $A$   
**from** *fr1b fr2* **have**  $M = -A \wedge \varphi A ; N = \varphi A$  **in**  $M \preceq N \longrightarrow (\varphi M \preceq N \vee \varphi N)$  **using** *PF1 PF6* **by** (*simp add: Fr-6-def*)  
**hence**  $\varphi(-A \wedge \varphi A) \preceq \varphi A \vee \varphi(\varphi A)$  **unfolding conn by meson**  
**hence**  $\varphi(-A \wedge \varphi A) \preceq \varphi A$  **using** *Fr-4-def fr4* **unfolding conn by metis**  
**hence** *aux:*  $\varphi(-A \wedge \varphi A) \wedge \neg(\varphi A) \approx \perp$  **unfolding conn by blast**  
**have**  $(\mathcal{B}_F \varphi)((\mathcal{B}_F \varphi)^d A) = \neg(-A \wedge \varphi(-A)) \wedge \varphi(\neg(-A \wedge \varphi(-A)))$  **unfolding** *Br-fr-def dual-def*  
**by** *simp*  
**also have**  $\dots = (A \vee \neg \varphi A) \wedge \varphi(-A \wedge \varphi A)$  **using** *Fr-2-def fr2* **unfolding conn by force**  
**also have**  $\dots = (A \wedge \varphi(-A \wedge \varphi A)) \vee (\neg \varphi A \wedge \varphi(-A \wedge \varphi A))$  **unfolding conn by auto**  
**also have**  $\dots = (A \wedge \varphi(-A \wedge \varphi A))$  **using** *aux* **unfolding conn by blast**  
**finally have**  $(\mathcal{B}_F \varphi)((\mathcal{B}_F \varphi)^d A) = A \wedge \varphi(-A \wedge \varphi A)$  **unfolding** *Br-fr-def dual-def conn* **by simp**  
**} thus**  $Br-3(\mathcal{B}_F \varphi)$  **unfolding** *Br-3-def Br-fr-def conn* **by meson**  
**qed**

Interior operator as derived from border.

**definition**  $Int-br::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma) (\mathcal{I}_B)$  **where**  $\mathcal{I}_B \mathcal{B} \equiv \lambda A. A \leftarrow (\mathcal{B} A)$   
**lemma** *IB1: Br-1*  $\varphi \Longrightarrow Int-1(\mathcal{I}_B \varphi)$  **using** *Br-1-def MULT-def Int-br-def conn* **by auto**  
**lemma** *IB2: Int-2*  $(\mathcal{I}_B \varphi)$  **by** (*simp add: dEXP-def Int-br-def conn*)  
**lemma** *IB3: Br-2*  $\varphi \Longrightarrow Int-3(\mathcal{I}_B \varphi)$  **by** (*simp add: Br-2-def dNOR-def Int-br-def conn*)  
**lemma** *IB4: Br-1*  $\varphi \Longrightarrow Br-3 \varphi \Longrightarrow Int-4(\mathcal{I}_B \varphi)$  **unfolding** *Int-br-def IDEM-def conn*  
**using** *Br-1-def Br-5c-def Br-5d-def PB5c PB5d dual-def* **unfolding conn by** (*metis (full-types)*)

Interior operator as derived from frontier.

**definition**  $Int\text{-}fr::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{I}_F$ ) **where**  $\mathcal{I}_F \mathcal{F} \equiv \lambda A. A \leftarrow (\mathcal{F} A)$

**lemma**  $IF1a: Fr\text{-}1b \varphi \Longrightarrow Int\text{-}1a(\mathcal{I}_F \varphi)$  **using**  $Fr\text{-}1b\text{-}def$   $MULT\text{-}a\text{-}def$   $Int\text{-}fr\text{-}def$  **conn** **by** *auto*

**lemma**  $IF1b: Fr\text{-}1a \varphi \Longrightarrow Int\text{-}1b(\mathcal{I}_F \varphi)$  **using**  $Fr\text{-}1a\text{-}def$   $MULT\text{-}b\text{-}def$   $Int\text{-}fr\text{-}def$  **conn** **by** *auto*

**lemma**  $IF1: Fr\text{-}1 \varphi \Longrightarrow Int\text{-}1(\mathcal{I}_F \varphi)$  **by** (*simp add: IF1a IF1b PF1 PI1*)

**lemma**  $IF2: Int\text{-}2(\mathcal{I}_F \varphi)$  **by** (*simp add: dEXP\text{-}def Int\text{-}fr\text{-}def conn*)

**lemma**  $IF3: Fr\text{-}2 \varphi \Longrightarrow Fr\text{-}3 \varphi \Longrightarrow Int\text{-}3(\mathcal{I}_F \varphi)$  **using**  $BF2$   $Br\text{-}2\text{-}def$   $Br\text{-}fr\text{-}def$   $dNOR\text{-}def$   $Int\text{-}fr\text{-}def$  **unfolding** **conn** **by** *auto*

**lemma**  $IF4: Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Fr\text{-}4 \varphi \Longrightarrow Int\text{-}4(\mathcal{I}_F \varphi)$

**using**  $Fr\text{-}1a\text{-}def$   $Fr\text{-}2\text{-}def$   $Fr\text{-}4\text{-}def$  **unfolding**  $Int\text{-}fr\text{-}def$   $IDEM\text{-}def$  **conn** **by** *metis*

Closure operator as derived from border.

**definition**  $Cl\text{-}br::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{C}_B$ ) **where**  $\mathcal{C}_B \mathcal{B} \equiv \lambda A. A \vee \mathcal{B}(-A)$

**lemma**  $CB1: Br\text{-}1 \varphi \Longrightarrow Cl\text{-}1(\mathcal{C}_B \varphi)$  **proof** –

**assume**  $br1: Br\text{-}1 \varphi$

{ **fix**  $A B$

**have**  $(\mathcal{C}_B \varphi) (A \vee B) = A \vee B \vee \varphi(-(A \vee B))$  **by** (*simp add: Cl\text{-}br\text{-}def conn*)

**also have**  $\dots = A \vee B \vee \varphi(-A \wedge -B)$  **unfolding** **conn** **by** *simp*

**also have**  $\dots = A \vee B \vee (-A \wedge \varphi(-B)) \vee (-B \wedge \varphi(-A))$  **using**  $Br\text{-}1\text{-}def$   $br1$  **unfolding** **conn**

**by** *auto*

**also have**  $\dots = A \vee \varphi(-A) \vee B \vee \varphi(-B)$  **unfolding** **conn** **by** *auto*

**also have**  $\dots = (\mathcal{C}_B \varphi) A \vee (\mathcal{C}_B \varphi) B$  **by** (*simp add: Cl\text{-}br\text{-}def conn*)

**finally have**  $(\mathcal{C}_B \varphi)(A \vee B) = (\mathcal{C}_B \varphi) A \vee (\mathcal{C}_B \varphi) B$  **unfolding**  $Cl\text{-}br\text{-}def$  **by** *blast*

} **thus**  $Cl\text{-}1(\mathcal{C}_B \varphi)$  **unfolding**  $ADDI\text{-}def$   $Cl\text{-}br\text{-}def$  **by** *simp*

**qed**

**lemma**  $CB2: Cl\text{-}2(\mathcal{C}_B \varphi)$  **by** (*simp add: EXP\text{-}def Cl\text{-}br\text{-}def conn*)

**lemma**  $CB3: Br\text{-}2 \varphi \Longrightarrow Cl\text{-}3(\mathcal{C}_B \varphi)$  **using**  $Br\text{-}2\text{-}def$   $Cl\text{-}br\text{-}def$   $IC3$   $dNOR\text{-}def$   $dual\text{-}def$   $dual\text{-}symm$  **unfolding** **conn** **by** *smt*

**lemma**  $CB4: Br\text{-}1 \varphi \Longrightarrow Br\text{-}3 \varphi \Longrightarrow Cl\text{-}4(\mathcal{C}_B \varphi)$  **proof** –

**assume**  $br1: Br\text{-}1 \varphi$  **and**  $br3: Br\text{-}3 \varphi$

{ **fix**  $A$

**have**  $(\mathcal{C}_B \varphi) ((\mathcal{C}_B \varphi) A) = A \vee \varphi(-A) \vee \varphi(-(A \vee \varphi(-A)))$  **by** (*simp add: Cl\text{-}br\text{-}def conn*)

**also have**  $\dots = A \vee \varphi(-A) \vee \varphi(-A \wedge \varphi^d A)$  **unfolding**  $dual\text{-}def$  **conn** **by** *simp*

**also have**  $\dots = A \vee \varphi(-A) \vee (-A \wedge \varphi(\varphi^d A)) \vee (-A \wedge \varphi(-A))$  **unfolding**  $dual\text{-}def$  **using**

$Br\text{-}1\text{-}def$   $br1$  **conn** **by** *auto*

**also have**  $\dots = A \vee \varphi(-A)$  **using**  $Br\text{-}3\text{-}def$   $br3$  **unfolding** **conn** **by** *metis*

**finally have**  $(\mathcal{C}_B \varphi) ((\mathcal{C}_B \varphi) A) = (\mathcal{C}_B \varphi) A$  **unfolding**  $Cl\text{-}br\text{-}def$  **by** *blast*

} **thus**  $Cl\text{-}4(\mathcal{C}_B \varphi)$  **unfolding**  $IDEM\text{-}def$   $Cl\text{-}br\text{-}def$  **by** *simp*

**qed**

Closure operator as derived from frontier.

**definition**  $Cl\text{-}fr::(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{C}_F$ ) **where**  $\mathcal{C}_F \mathcal{F} \equiv \lambda A. A \vee (\mathcal{F} A)$

**lemma**  $CF1b: Fr\text{-}1b \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Cl\text{-}1b(\mathcal{C}_F \varphi)$  **using**  $Cl\text{-}fr\text{-}def$   $Fr\text{-}6\text{-}def$   $MONO\text{-}def$   $MONO\text{-}ADDIb$   $PF6$  **unfolding** **conn** **by** *auto*

**lemma**  $CF1a: Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Cl\text{-}1a(\mathcal{C}_F \varphi)$  **proof** –

**have**  $aux: Fr\text{-}2 \varphi \Longrightarrow (\mathcal{I}_F \varphi)^d = (\mathcal{C}_F \varphi)$  **by** (*simp add: Cl\text{-}fr\text{-}def Fr\text{-}2\text{-}def Int\text{-}fr\text{-}def dual\text{-}def conn*)

**hence**  $Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Cl\text{-}1a(\mathcal{I}_F \varphi)^d$  **using**  $IC1b$   $IF1b$  **by** *blast*

**thus**  $Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Cl\text{-}1a(\mathcal{C}_F \varphi)$  **using**  $ADDI\text{-}a\text{-}def$   $aux$  **unfolding** **conn** **by** *simp*

**qed**

**lemma**  $CF1: Fr\text{-}1 \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Cl\text{-}1(\mathcal{C}_F \varphi)$  **by** (*simp add: CF1a CF1b PC1 PF1*)

**lemma**  $CF2: Cl\text{-}2(\mathcal{C}_F \varphi)$  **by** (*simp add: EXP\text{-}def Cl\text{-}fr\text{-}def conn*)

**lemma**  $CF3: Fr\text{-}3 \varphi \Longrightarrow Cl\text{-}3(\mathcal{C}_F \varphi)$  **by** (*simp add: Cl\text{-}fr\text{-}def NOR\text{-}def conn*)

**lemma**  $CF4: Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Fr\text{-}4 \varphi \Longrightarrow Cl\text{-}4(\mathcal{C}_F \varphi)$  **proof** –

**have**  $aux: Fr\text{-}2 \varphi \Longrightarrow (\mathcal{I}_F \varphi)^d = (\mathcal{C}_F \varphi)$  **by** (*simp add: Cl\text{-}fr\text{-}def Fr\text{-}2\text{-}def Int\text{-}fr\text{-}def dual\text{-}def conn*)

**hence**  $Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Fr\text{-}4 \varphi \Longrightarrow Cl\text{-}4(\mathcal{I}_F \varphi)^d$  **using**  $IC4$   $IF4$  **by** *blast*

**thus**  $Fr\text{-}1a \varphi \Longrightarrow Fr\text{-}2 \varphi \Longrightarrow Fr\text{-}4 \varphi \Longrightarrow Cl\text{-}4(\mathcal{C}_F \varphi)$  **by** (*simp add: aux*)

qed

## 4.2.4 Infinitary conditions

We define the essential infinitary conditions for the closure and interior operators (entailing infinite additivity and multiplicativity resp.). Observe that the other direction is implied by monotonicity (MONO).

**abbreviation**  $Cl\text{-}inf \ \varphi \equiv iADDI\text{-}a(\varphi)$

**abbreviation**  $Int\text{-}inf \ \varphi \equiv iMULT\text{-}b(\varphi)$

There exists indeed a condition on frontier operators responsible for the infinitary conditions above:

**definition**  $Fr\text{-}inf \ \varphi \equiv \forall S. \bigwedge S \wedge \varphi(\bigwedge S) \preceq \bigwedge S \wedge \bigvee Ra[\varphi|S]$

**lemma**  $CF\text{-}inf: Fr\text{-}2 \ \varphi \implies Fr\text{-}inf \ \varphi \implies Cl\text{-}inf(C_F \ \varphi)$  **unfolding**  $iADDI\text{-}a\text{-}def \ Fr\text{-}inf\text{-}def$

**by** ( $smt \ Cl\text{-}fr\text{-}def \ Fr\text{-}2\text{-}def \ Ra\text{-}restr\text{-}ex \ compl\text{-}def \ dom\text{-}compl\text{-}def2 \ iDM\text{-}b \ join\text{-}def \ meet\text{-}def \ supremum\text{-}def$ )

**lemma**  $IF\text{-}inf: Fr\text{-}inf \ \varphi \implies Int\text{-}inf(\mathcal{I}_F \ \varphi)$  **unfolding**  $Fr\text{-}inf\text{-}def \ iMULT\text{-}b\text{-}def \ Int\text{-}fr\text{-}def \ Ra\text{-}restr\text{-}all$

**by** ( $metis \ (mono\text{-}tags, \ hide\text{-}lams) \ diff\text{-}def \ infimum\text{-}def \ meet\text{-}def \ pfunRange\text{-}restr\text{-}def \ supremum\text{-}def$ )

This condition is indeed strong enough to entail closure of the fixed-point predicates under infimum/supremum.

**lemma**  $fp\text{-}IF\text{-}inf\text{-}closed: Fr\text{-}inf \ \varphi \implies infimum\text{-}closed \ (fp \ (\mathcal{I}_F \ \varphi))$  **by** ( $metis \ (full\text{-}types) \ IF2 \ IF\text{-}inf \ Ra\text{-}restr\text{-}all \ dEXP\text{-}def \ iMULT\text{-}b\text{-}def \ infimum\text{-}def$ )

**lemma**  $fp\text{-}CF\text{-}sup\text{-}closed: Fr\text{-}inf \ \varphi \implies Fr\text{-}2 \ \varphi \implies supremum\text{-}closed \ (fp \ (C_F \ \varphi))$  **by** ( $metis \ (full\text{-}types) \ CF2 \ CF\text{-}inf \ EXP\text{-}def \ Ra\text{-}restr\text{-}ex \ iADDI\text{-}a\text{-}def \ supremum\text{-}def$ )

end

**theory**  $topo\text{-}operators\text{-}derivative$

**imports**  $topo\text{-}operators\text{-}basic$

**begin**

**nitpick\text{-}params**[ $assms=true, \ user\text{-}axioms=true, \ show\text{-}all, \ expect=genuine, \ format=3$ ]

## 4.3 Derivative and coherence

In this section we investigate two related operators, namely the ‘derivative’ (or ‘derived set’) and the (Cantorian) ‘coherence’ of a set. The derivative of a set is the set of its accumulation (aka. limit) points. The coherence of a set A is the set formed by those limit points of A belonging to A. For the derivative operator we draw upon the works by Kuratowski [8] and (in more detail) by Zarycki [14]; cf. also McKinsey & Tarski [11]. For the (Cantorian) coherence operator we follow the treatment given by Zarycki in [13].

### 4.3.1 Derivative conditions

The derivative conditions overlap partly with Kuratowski closure conditions [8]. We try to make both notations coincide when possible.

**abbreviation**  $Der\text{-}1 \ \varphi \equiv Cl\text{-}1 \ \varphi$

**abbreviation**  $Der\text{-}1a \ \varphi \equiv Cl\text{-}1a \ \varphi$

**abbreviation**  $Der\text{-}1b \ \varphi \equiv Cl\text{-}1b \ \varphi$

**abbreviation**  $Der\text{-}2 \ \varphi \equiv Cl\text{-}5 \ \varphi$  — follows from Cl-2

**abbreviation**  $Der\text{-}3 \ \varphi \equiv Cl\text{-}3 \ \varphi$

**abbreviation**  $Der\text{-}4 \ \varphi \equiv Cl\text{-}4' \ \varphi$

**definition**  $Der\text{-}4e \ \varphi \equiv \forall A. \varphi(\varphi A) \preceq (\varphi A \vee A)$

**definition** *Der-5*  $\varphi \equiv \forall A. (\varphi A \preceq A) \wedge (A \preceq \varphi^d A) \longrightarrow (A \approx \perp \vee A \approx \top)$

Some remarks: Condition *Der-2* basically says (when assuming other derivative axioms) that the space is dense-in-itself, i.e. that all points are accumulation points (no point is isolated) w.r.t the whole space. *Der-4* is a weakened (left-to-right) variant of *Cl-4*. Condition *Der-4e* corresponds to a (weaker) condition than *Der-4* and is used in more recent literature (in particular in the works of Leo Esakia [5]). When other derivative axioms are assumed, *Der-5* above as used by Zarycki [14] says that the only clopen sets in the space are the top and bottom elements (empty set and universe, resp.). We verify some properties:

**lemma** *Der4e-rel*: *Der-4*  $\varphi \implies$  *Der-4e*  $\varphi$  **by** (*simp add: IDEMb-def Der-4e-def conn*)

**lemma** *PD1*: *Der-1b*  $\varphi \implies \forall A B. A \preceq B \longrightarrow \varphi A \preceq \varphi B$  **using** *MONO-def MONO-ADDIb* **by** *auto*

**lemma** *PD2*: *Der-1b*  $\varphi \implies \forall A B. A \preceq B \longrightarrow \varphi^d A \preceq \varphi^d B$  **using** *CI1b MONO-def MONO-MULTa dual-def* **by** *fastforce*

**lemma** *PD3*: *Der-1b*  $\varphi \implies \forall A B. \varphi(A \wedge B) \preceq \varphi A \wedge \varphi B$  **unfolding** *conn* **by** (*metis (no-types, lifting) PD1*)

**lemma** *PD4*: *Der-1*  $\varphi \implies \forall A B. (\varphi A \leftarrow \varphi B) \preceq \varphi(A \leftarrow B)$  **using** *Cl-6-def PC6* **by** *metis*

**lemma** *PD5*: *Der-4*  $\varphi \implies \forall A. \varphi(\varphi(-(\varphi A))) \preceq \varphi(-(\varphi A))$  **unfolding** *IDEMb-def* **by** *blast*

Observe that the lemmas below require *Der-2* as premise:

**lemma** *PD6*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies \forall A. \varphi^d A \preceq \varphi A$  **unfolding** *ADDI-def dNOR-def dual-def conn* **by** *fastforce*

**lemma** *PD7*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies \forall A. \varphi(\varphi^d A) \preceq \varphi(\varphi A)$  **by** (*metis (mono-tags, lifting) PC1 PD1 PD6*)

**lemma** *PD8*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies$  *Der-4*  $\varphi \implies \forall A. \varphi(\varphi^d A) \preceq \varphi A$  **by** (*meson IDEMb-def PD7*)

**lemma** *PD9*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies$  *Der-4*  $\varphi \implies \forall A. \varphi^d A \preceq \varphi^d(\varphi A)$  **by** (*metis CI4' IDEMa-def PC1 PD2 PD6*)

**lemma** *PD10*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies$  *Der-4*  $\varphi \implies \forall A. \varphi^d A \preceq \varphi(\varphi^d A)$  **using** *CI4' IDEMa-def PD6* **by** *metis*

**lemma** *PD11*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies$  *Der-4*  $\varphi \implies \forall A. -(\varphi A) \preceq \varphi(-(\varphi A))$  **using** *IDEMb-def PD6 dual-def* **unfolding** *conn* **by** *metis*

**lemma** *PD12*: *Der-1*  $\varphi \implies$  *Der-2*  $\varphi \implies$  *Der-4*  $\varphi \implies \forall A. (\varphi^d A) \wedge (\varphi A) \approx \varphi^d(A \wedge (\varphi A))$  **proof**

—  
**assume** *der1*: *Der-1*  $\varphi$  **and** *der2*: *Der-2*  $\varphi$  **and** *der4*: *Der-4*  $\varphi$   
{ **fix**  $A$   
**from** *der1* **have**  $let\ M=-A ; N=-(\varphi A)$  **in**  $\varphi(M \vee N) \approx (\varphi M) \vee (\varphi N)$  **unfolding** *ADDI-def*  
**by** *simp*  
**hence** *aux*:  $-(\varphi(-A) \vee \varphi(-(\varphi A))) \approx -(\varphi(-A \vee -(\varphi A)))$  **unfolding** *dual-def conn* **by** *simp*  
**have**  $(\varphi^d A \wedge \varphi A) = (\varphi^d A \wedge \varphi^d(\varphi A))$  **using** *PD6 PD9 der1 der2 der4* **unfolding** *conn* **by** *metis*  
**also** **have**  $\dots = -(\varphi(-A) \vee \varphi(-(\varphi A)))$  **unfolding** *dual-def conn* **by** *simp*  
**also** **from** *aux* **have**  $\dots = -(\varphi(-A \vee -(\varphi A)))$  **unfolding** *dual-def* **by** *blast*  
**also** **have**  $\dots = \varphi^d(A \wedge (\varphi A))$  **unfolding** *dual-def conn* **by** *simp*  
**finally** **have**  $(\varphi^d A) \wedge (\varphi A) \approx \varphi^d(A \wedge (\varphi A))$  **by** *simp*  
} **thus** *?thesis* **by** *simp*  
**qed**

The conditions below can serve to axiomatize a derivative operator. Different authors consider different sets of conditions. We define below some corresponding to Zarycki [14], Kuratowski [8] [13], McKinsey & Tarski [11], and Esakia [5], respectively.

**abbreviation**  $\mathfrak{D}z \varphi \equiv$  *Der-1*  $\varphi \wedge$  *Der-2*  $\varphi \wedge$  *Der-3*  $\varphi \wedge$  *Der-4*  $\varphi \wedge$  *Der-5*  $\varphi$

**abbreviation**  $\mathfrak{D}k \varphi \equiv$  *Der-1*  $\varphi \wedge$  *Der-2*  $\varphi \wedge$  *Der-3*  $\varphi \wedge$  *Der-4*  $\varphi$

**abbreviation**  $\mathfrak{D}mt \varphi \equiv$  *Der-1*  $\varphi \wedge$  *Der-3*  $\varphi \wedge$  *Der-4*  $\varphi$

**abbreviation**  $\mathfrak{D}e \varphi \equiv$  *Der-1*  $\varphi \wedge$  *Der-3*  $\varphi \wedge$  *Der-4e*  $\varphi$



Our ‘default’ derivative operator will coincide with  $\mathfrak{D}k$  from Kuratowski (also Zarycki). However, for proving theorems we will employ the weaker variant Der-4e instead of Der-4 whenever possible. We start by defining a dual operator and verifying some dualities; we then define conversion operators. Observe that conditions Der-2 and Der-5 are not used in the rest of this subsection. Der-2 will be required later when working with the coherence operator.

**abbreviation**  $\mathfrak{D} \varphi \equiv \mathfrak{D}k \varphi$

**abbreviation**  $\Sigma \varphi \equiv \text{Int-1 } \varphi \wedge \text{Int-3 } \varphi \wedge \text{Int-4}' \varphi \wedge \text{Int-5 } \varphi$  — ‘dual-derivative’ operator

**lemma** *SD-dual1*:  $\Sigma(\varphi) \implies \mathfrak{D}(\varphi^d)$  **using** *IC1 IC4' IC3 IC5* **by** *simp*

**lemma** *SD-dual2*:  $\Sigma(\varphi^d) \implies \mathfrak{D}(\varphi)$  **using** *IC1 IC4' IC3 IC5 dual-symm* **by** *metis*

**lemma** *DS-dual1*:  $\mathfrak{D}(\varphi) \implies \Sigma(\varphi^d)$  **using** *CI1 CI4' CI3 CI5* **by** *simp*

**lemma** *DS-dual2*:  $\mathfrak{D}(\varphi^d) \implies \Sigma(\varphi)$  **using** *CI1 CI4' CI3 CI5 dual-symm* **by** *metis*

**lemma** *DS-dual*:  $\mathfrak{D}(\varphi) = \Sigma(\varphi^d)$  **using** *SD-dual2 DS-dual1* **by** *smt*

Closure operator as derived from derivative.

**definition** *Cl-der*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{C}_D \mathcal{D}$ ) **where**  $\mathcal{C}_D \mathcal{D} \equiv \lambda A. A \vee \mathcal{D}(A)$

Verify properties:

**lemma** *CD1a*: *Der-1a*  $\mathcal{D} \implies \text{Cl-1a}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def ADDI-a-def conn* **by** *auto*

**lemma** *CD1b*: *Der-1b*  $\mathcal{D} \implies \text{Cl-1b}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def ADDI-b-def conn* **by** *auto*

**lemma** *CD1* : *Der-1*  $\mathcal{D} \implies \text{Cl-1}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def ADDI-def conn* **by** *blast*

**lemma** *CD2*: *Cl-2* ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def EXP-def conn* **by** *simp*

**lemma** *CD3*: *Der-3*  $\mathcal{D} \implies \text{Der-3}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def NOR-def conn* **by** *simp*

**lemma** *CD4a*: *Der-1a*  $\mathcal{D} \implies \text{Der-4e}$   $\mathcal{D} \implies \text{Cl-4}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def* **by** (*smt ADDI-a-def Der-4e-def IDEM-def join-def*)

**lemma** *Der-1b*  $\mathcal{D} \implies \text{Der-4}$   $\mathcal{D} \implies \text{Cl-4}$  ( $\mathcal{C}_D \mathcal{D}$ ) **nitpick oops**

**lemma** *CD4*: *Der-1*  $\mathcal{D} \implies \text{Der-4e}$   $\mathcal{D} \implies \text{Cl-4}$  ( $\mathcal{C}_D \mathcal{D}$ ) **unfolding** *Cl-der-def* **by** (*metis (no-types, lifting) CD4a Cl-der-def IDEM-def PC1*)

Interior operator as derived from (dual) derivative.

**definition** *Int-der*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{I}_D \mathcal{D}$ ) **where**  $\mathcal{I}_D \mathcal{D} \equiv \lambda A. A \wedge \mathcal{D}^d(A)$

Verify definition:

**lemma** *Int-der-def2*:  $\mathcal{I}_D \mathcal{D} = (\lambda \varphi. \varphi \leftarrow \mathcal{D}(-\varphi))$  **unfolding** *Int-der-def dual-def conn* **by** *simp*

**lemma** *dual-der1*:  $\mathcal{C}_D \mathcal{D} \equiv (\mathcal{I}_D \mathcal{D})^d$  **unfolding** *Cl-der-def Int-der-def dual-def conn* **by** *simp*

**lemma** *dual-der2*:  $\mathcal{I}_D \mathcal{D} \equiv (\mathcal{C}_D \mathcal{D})^d$  **unfolding** *Cl-der-def Int-der-def dual-def conn* **by** *simp*

Verify properties:

**lemma** *ID1*: *Der-1*  $\mathcal{D} \implies \text{Int-1}$  ( $\mathcal{I}_D \mathcal{D}$ ) **using** *Int-der-def MULT-def ADDI-def CI1* **unfolding** *conn* **by** *auto*

**lemma** *ID1a*: *Der-1a*  $\mathcal{D} \implies \text{Int-1b}$  ( $\mathcal{I}_D \mathcal{D}$ ) **by** (*simp add: CI1a dual-der2 CD1a*)

**lemma** *ID1b*: *Der-1b*  $\mathcal{D} \implies \text{Int-1a}$  ( $\mathcal{I}_D \mathcal{D}$ ) **by** (*simp add: CI1b dual-der2 CD1b*)

**lemma** *ID2*: *Int-2* ( $\mathcal{I}_D \mathcal{D}$ ) **unfolding** *Int-der-def dEXP-def conn* **by** *simp*

**lemma** *ID3*: *Der-3*  $\mathcal{D} \implies \text{Int-3}$  ( $\mathcal{I}_D \mathcal{D}$ ) **by** (*simp add: CI3 CD3 dual-der2*)

**lemma** *ID4*: *Der-1*  $\mathcal{D} \implies \text{Der-4e}$   $\mathcal{D} \implies \text{Int-4}$  ( $\mathcal{I}_D \mathcal{D}$ ) **using** *CI4 CD4 dual-der2* **by** *simp*

**lemma** *ID4a*: *Der-1a*  $\mathcal{D} \implies \text{Der-4e}$   $\mathcal{D} \implies \text{Int-4}$  ( $\mathcal{I}_D \mathcal{D}$ ) **by** (*simp add: CI4 dual-der2 CD4a*)

**lemma** *Der-1b*  $\mathcal{D} \implies \text{Der-4}$   $\mathcal{D} \implies \text{Int-4}$  ( $\mathcal{I}_D \mathcal{D}$ ) **nitpick oops**

Border operator as derived from (dual) derivative.

**definition** *Br-der*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{B}_D \mathcal{D}$ ) **where**  $\mathcal{B}_D \mathcal{D} \equiv \lambda A. A \leftarrow \mathcal{D}^d(A)$

Verify definition:

**lemma** *Br-der-def2*:  $\mathcal{B}_D \mathcal{D} = (\lambda A. A \wedge \mathcal{D}(-A))$  **unfolding** *Br-der-def dual-def conn* **by** *simp*

Verify properties:

**lemma** *BD1*:  $Der-1 \mathcal{D} \implies Br-1 (\mathcal{B}_D \mathcal{D})$  **using** *Br-der-def Br-1-def CI1 MULT-def conn* **by** *auto*

**lemma** *BD2*:  $Der-3 \mathcal{D} \implies Br-2 (\mathcal{B}_D \mathcal{D})$  **using** *Br-der-def Br-2-def CI3 dNOR-def* **unfolding** *conn* **by** *auto*

**lemma** *BD3*:  $Der-1b \mathcal{D} \implies Der-4e \mathcal{D} \implies Br-3 (\mathcal{B}_D \mathcal{D})$  **using** *dual-def Der-4e-def MONO-ADDIb MONO-def* **unfolding** *Br-der-def Br-3-def conn* **by** *smt*

Frontier operator as derived from derivative.

**definition** *Fr-der*:: $(\sigma \implies \sigma) \implies (\sigma \implies \sigma)$  ( $\mathcal{F}_D$ ) **where**  $\mathcal{F}_D \mathcal{D} \equiv \lambda A. (A \leftarrow \mathcal{D}^d(A)) \vee (\mathcal{D}(A) \leftarrow A)$

Verify definition:

**lemma** *Fr-der-def2*:  $\mathcal{F}_D \mathcal{D} = (\lambda A. (A \vee \mathcal{D}(A)) \wedge (\neg A \vee \mathcal{D}(\neg A)))$  **unfolding** *Fr-der-def dual-def conn* **by** *auto*

Verify properties:

**lemma** *FD1a*:  $Der-1a \mathcal{D} \implies Fr-1a(\mathcal{F}_D \mathcal{D})$  **unfolding** *Fr-1a-def Fr-der-def* **using** *CI1a MULT-b-def conn* **by** *auto*

**lemma** *FD1b*:  $Der-1b \mathcal{D} \implies Fr-1b(\mathcal{F}_D \mathcal{D})$  **unfolding** *Fr-1b-def Fr-der-def* **using** *CI1b MULT-a-def conn* **by** *smt*

**lemma** *FD1*:  $Der-1 \mathcal{D} \implies Fr-1(\mathcal{F}_D \mathcal{D})$  **unfolding** *Fr-1-def Fr-der-def* **using** *CI1 MULT-def conn* **by** *auto*

**lemma** *FD2*:  $Fr-2(\mathcal{F}_D \mathcal{D})$  **using** *dual-def dual-symm* **unfolding** *Fr-2-def Fr-der-def conn* **by** *metis*

**lemma** *FD3*:  $Der-3 \mathcal{D} \implies Fr-3(\mathcal{F}_D \mathcal{D})$  **by** (*simp add: NOR-def Fr-der-def conn*)

**lemma** *FD4*:  $Der-1 \mathcal{D} \implies Der-4e \mathcal{D} \implies Fr-4(\mathcal{F}_D \mathcal{D})$  **by** (*metis CD1b CD2 CD4 Cl-8-def Cl-der-def Fr-4-def Fr-der-def2 PC1 PC8 meet-def*)

Note that the derivative operation cannot be obtained from interior, closure, border, nor frontier. In this respect the derivative is a fundamental operation.

### 4.3.2 Infinitary conditions

The corresponding infinitary condition on derivative operators is inherited from the one for closure.

**abbreviation** *Der-inf*  $\varphi \equiv Cl-inf(\varphi)$

**lemma** *CD-inf*:  $Der-inf \varphi \implies Cl-inf(\mathcal{C}_D \varphi)$  **unfolding** *iADDI-a-def* **by** (*smt Cl-der-def Fr-2-def Ra-restr-ex compl-def dom-compl-def2 iDM-b join-def meet-def supremum-def*)

**lemma** *ID-inf*:  $Der-inf \varphi \implies Int-inf(\mathcal{I}_D \varphi)$  **by** (*simp add: CD-inf dual-der2 iADDI-MULT-dual1*)

This condition is indeed strong enough as to entail closure of some fixed-point predicates under infimum/supremum.

**lemma** *fp-ID-inf-closed*:  $Der-inf \varphi \implies infimum-closed (fp (\mathcal{I}_D \varphi))$  **by** (*metis (full-types) ID2 ID-inf Ra-restr-all dEXP-def iMULT-b-def inf-char*)

**lemma** *fp-CD-sup-closed*:  $Der-inf \varphi \implies supremum-closed (fp (\mathcal{C}_D \varphi))$  **by** (*metis (full-types) CD-inf Cl-der-def Ra-restr-ex iADDI-a-def join-def supremum-def*)

### 4.3.3 Coherence conditions

We finish this section by introducing the ‘coherence’ operator (Cantor’s ‘Koherenz’) as discussed by Zarycki in [13]. As happens with the derivative operator, the coherence operator cannot be derived from interior, closure, border, nor frontier.

**definition** *Kh-1*  $\varphi \equiv ADDI-b \varphi$

**definition** *Kh-2*  $\varphi \equiv dEXP \varphi$

**definition** *Kh-3*  $\varphi \equiv \forall A. \varphi(\varphi^d A) \approx \varphi^d(\varphi A)$

**lemma** *PK1*: *Kh-1*  $\varphi \equiv MONO \varphi$  **using** *ADDI-b-def Kh-1-def MONO-ADDIb* **by** *auto*

**lemma** *PK2*: *Kh-1*  $\varphi \equiv \forall A B. \varphi(A \wedge B) \preceq (\varphi A) \wedge (\varphi B)$  **using** *MULT-a-def MONO-MULTa PK1* **by** *auto*

**lemma** *PK3*: *Kh-2*  $\varphi \implies \varphi \perp \approx \perp$  **using** *Kh-2-def dEXP-def unfolding conn* **by** *auto*

**lemma** *PK4*: *Kh-1*  $\varphi \implies Kh-3 \varphi \implies \varphi \top \approx \top$  **using** *MONO-def PK1 dual-def unfolding Kh-3-def conn* **by** *metis*

**lemma** *PK5*: *Kh-2*  $\varphi \implies \forall A. \varphi(-A) \preceq -(\varphi A)$  **unfolding** *Kh-2-def dEXP-def conn* **by** *metis*

**lemma** *PK6*: *Kh-1*  $\varphi \implies Kh-2 \varphi \implies \forall A B. \varphi(A \leftarrow B) \preceq (\varphi A) \leftarrow (\varphi B)$  **unfolding** *conn* **by** *(metis (no-types, lifting) Kh-2-def dEXP-def MONO-def PK1)*

**lemma** *PK7*: *Kh-3*  $\varphi \implies \forall A. \varphi(\varphi(-(\varphi A))) \approx \varphi(-(\varphi(\varphi^d A)))$  **proof** –

**assume** *kh3*: *Kh-3*  $\varphi$

{ **fix** *A*

**from** *kh3* **have**  $\varphi(\varphi^d A) = \varphi^d(\varphi A)$  **using** *Kh-3-def* **by** *auto*

**hence**  $\varphi(-(\varphi(\varphi^d A))) \approx \varphi(\varphi(-(\varphi A)))$  **unfolding** *dual-def conn* **by** *simp*

} **thus** *?thesis* **by** *simp*

**qed**

**lemma** *PK8*: *Kh-3*  $\varphi \implies \forall A. \varphi(-(\varphi(\varphi A))) \approx \varphi^d(\varphi(-(\varphi A)))$  **proof** –

**assume** *kh3*: *Kh-3*  $\varphi$

{ **fix** *A*

**from** *kh3* **have** *aux*:  $\varphi^d(\varphi A) \approx \varphi(\varphi^d A)$  **using** *Kh-3-def* **by** *simp*

**have** *let*  $A = -(\varphi A)$  **in**  $(\varphi^d(\varphi A) \approx \varphi(\varphi^d A))$  **using** *Kh-3-def kh3* **by** *auto*

**hence**  $\varphi(-(\varphi(\varphi A))) \approx \varphi^d(\varphi(-(\varphi A)))$  **unfolding** *dual-def conn* **by** *simp*

} **thus** *?thesis* **by** *simp*

**qed**

Coherence operator as derived from derivative (requires conditions Der-2 and Der-4).

**definition** *Kh-der*:: $(\sigma \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{K}_D$ ) **where**  $\mathcal{K}_D \mathcal{D} \equiv \lambda A. A \wedge (\mathcal{D} A)$

Verify properties:

**lemma** *KD1*: *Der-1*  $\varphi \implies Kh-1$  ( $\mathcal{K}_D \varphi$ ) **using** *PC1 PD3 PK2 ADDI-def Kh-der-def* **unfolding** *conn* **by** *(metis (full-types))*

**lemma** *KD2*: *Kh-2* ( $\mathcal{K}_D \varphi$ ) **by** *(simp add: Kh-2-def dEXP-def Kh-der-def conn)*

**lemma** *KD3*: *Der-1*  $\varphi \implies Der-2 \varphi \implies Der-4 \varphi \implies Kh-3$  ( $\mathcal{K}_D \varphi$ ) **proof** –

**assume** *der1*: *Der-1*  $\varphi$  **and** *der2*: *Der-2*  $\varphi$  **and** *der4*: *Der-4*  $\varphi$

{ **fix** *A*

**from** *der1* **have** *aux1*: *let*  $M = A$  ;  $N = (\varphi^d A)$  **in**  $\varphi(M \vee N) \approx (\varphi M \vee \varphi N)$  **unfolding** *ADDI-def* **by** *simp*

**from** *der1 der2 der4* **have** *aux2*:  $(\varphi^d A) \wedge (\varphi A) \approx \varphi^d(A \wedge \varphi A)$  **using** *PD12* **by** *simp*

**have**  $(\mathcal{K}_D \varphi)((\mathcal{K}_D \varphi)^d A) = (-(-A \wedge \varphi(-A)) \wedge \varphi(-(-A \wedge \varphi(-A))))$  **unfolding** *Kh-der-def dual-def* **by** *simp*

**also** **have**  $\dots = (A \vee \varphi^d A) \wedge \varphi(A \vee \varphi^d A)$  **unfolding** *dual-def conn* **by** *simp*

**also** **from** *aux1* **have**  $\dots = (A \vee \varphi^d A) \wedge (\varphi A \vee \varphi(\varphi^d A))$  **unfolding** *conn* **by** *meson*

**also** **have**  $\dots = (A \wedge \varphi A) \vee \varphi^d A$  **using** *PD6 PD8 der1 der2 der4* **unfolding** *conn* **by** *blast*

**also** **have**  $\dots = (A \wedge \varphi A) \vee (\varphi^d A \wedge \varphi A)$  **using** *PD6 der1 der2* **unfolding** *conn* **by** *blast*

**also** **have**  $\dots = (A \vee \varphi^d A) \wedge (\varphi A)$  **unfolding** *conn* **by** *auto*

**also** **from** *aux2* **have**  $\dots = (A \wedge \varphi A) \vee \varphi^d(A \wedge \varphi A)$  **unfolding** *dual-def conn* **by** *meson*

**also** **have**  $\dots = -(- (A \wedge \varphi A) \wedge \varphi(- (A \wedge \varphi A)))$  **unfolding** *dual-def conn* **by** *simp*

**also** **have**  $\dots = (\mathcal{K}_D \varphi)^d((\mathcal{K}_D \varphi) A)$  **unfolding** *Kh-der-def dual-def* **by** *simp*

**finally** **have**  $(\mathcal{K}_D \varphi)((\mathcal{K}_D \varphi)^d A) \approx (\mathcal{K}_D \varphi)^d((\mathcal{K}_D \varphi) A)$  **by** *simp*

} **thus** *?thesis* **unfolding** *Kh-3-def* **by** *simp*

**qed**

```

end
theory topo-alexandrov
  imports sse-operation-positive-quantification
begin
nitpick-params[assms=true, user-axioms=true, show-all, expect=genuine, format=3]

```

## 5 Generalized specialization orderings and Alexandrov topologies

A topology is called 'Alexandrov' (after the Russian mathematician Pavel Alexandrov) if the intersection (resp. union) of any (finite or infinite) family of open (resp. closed) sets is open (resp. closed); in algebraic terms, this means that the set of fixed points of the interior (closure) operation is closed under infinite meets (joins). Another common algebraic formulation requires the closure (interior) operation to satisfy the infinitary variants of additivity (multiplicativity), i.e. iADDI (iMULT) as introduced before.

In the literature, the well-known Kuratowski conditions for the closure (resp. interior) operation are assumed, namely: ADDI, EXP, NOR, IDEM (resp. MULT, dEXP, dNOR, IDEM). This makes both formulations equivalent. However, this is not the case in general if those conditions become negotiable.

Alexandrov topologies have interesting properties relating them to the semantics of modal logic. Assuming Kuratowski conditions, Alexandrov topological operations defined on subsets of  $S$  are in one-to-one correspondence with preorders on  $S$ ; in topological terms, Alexandrov topologies are uniquely determined by their specialization preorders. Since we do not presuppose any Kuratowski conditions to begin with, the preorders in question are in general not even transitive. Here we just call them 'specialization relations'. We will still call (generalized) closure/interior-like operations as such (for lack of a better name). We explore minimal conditions under which some relevant results for the semantics of modal logic obtain.

### 5.1 Specialization relations

Specialization relations (among worlds/points) are particular cases of propositional functions with type  $w \Rightarrow \sigma$ .

Define some relevant properties of relations:

**abbreviation** *serial*  $R \equiv \forall x. \exists y. R x y$   
**abbreviation** *reflexive*  $R \equiv \forall x. R x x$   
**abbreviation** *transitive*  $R \equiv \forall x y z. R x y \wedge R y z \longrightarrow R x z$   
**abbreviation** *antisymmetric*  $R \equiv \forall x y. R x y \wedge R y x \longrightarrow x = y$   
**abbreviation** *symmetric*  $R \equiv \forall x y. R x y \longrightarrow R y x$

Closure/interior operations can be derived from an arbitrary relation as operations returning down-/up-sets.

**definition** *Cl-rel*:: $(w \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{C}_R$ ) **where**  $\mathcal{C}_R R \equiv \lambda A. \lambda w. \exists v. R w v \wedge A v$   
**definition** *Int-rel*:: $(w \Rightarrow \sigma) \Rightarrow (\sigma \Rightarrow \sigma)$  ( $\mathcal{I}_R$ ) **where**  $\mathcal{I}_R R \equiv \lambda A. \lambda w. \forall v. R w v \longrightarrow A v$

Duality between interior and closure follows directly:

**lemma** *dual-rel1*:  $\forall A. (\mathcal{C}_R R) A \approx (\mathcal{I}_R R)^d A$  **unfolding** *Cl-rel-def Int-rel-def dual-def conn* **by** *simp*  
**lemma** *dual-rel2*:  $\forall A. (\mathcal{I}_R R) A \approx (\mathcal{C}_R R)^d A$  **unfolding** *Cl-rel-def Int-rel-def dual-def conn* **by** *simp*

We explore minimal conditions of the specialization relation under which some operation's conditions obtain.

**lemma** *rC1*:  $\text{ADDI } (\mathcal{C}_R R)$  **unfolding** *Cl-rel-def ADDI-def conn by blast*  
**lemma** *rC1i*:  $i\text{ADDI } (\mathcal{C}_R R)$  **by** (*smt Cl-rel-def Ra-restr-ex iADDI-def supremum-def*)  
**lemma** *rC2*:  $\text{reflexive } R \longrightarrow \text{EXP } (\mathcal{C}_R R)$  **unfolding** *EXP-def Cl-rel-def by auto*  
**lemma** *rC3*:  $\text{NOR } (\mathcal{C}_R R)$  **unfolding** *Cl-rel-def NOR-def conn by blast*  
**lemma** *rC4*:  $\text{reflexive } R \wedge \text{transitive } R \longrightarrow \text{IDEM } (\mathcal{C}_R R)$  **unfolding** *Cl-rel-def IDEM-def by smt*  
**lemma** *rC-Barcan*:  $\forall \pi. (\mathcal{C}_R R)(\exists x. \pi x) \preceq (\exists x. (\mathcal{C}_R R)(\pi x))$  **unfolding** *Cl-rel-def by auto*

**lemma** *rI1*:  $\text{MULT } (\mathcal{I}_R R)$  **unfolding** *Int-rel-def MULT-def conn by blast*  
**lemma** *rI1i*:  $i\text{MULT } (\mathcal{I}_R R)$  **by** (*smt Int-rel-def Ra-restr-all iMULT-def infimum-def*)  
**lemma** *rI2*:  $\text{reflexive } R \Longrightarrow \text{dEXP } (\mathcal{I}_R R)$  **unfolding** *Int-rel-def dEXP-def Int-rel-def by auto*  
**lemma** *rI3*:  $\text{dNOR } (\mathcal{I}_R R)$  **unfolding** *Int-rel-def dNOR-def conn by simp*  
**lemma** *rI4*:  $\text{reflexive } R \Longrightarrow \text{transitive } R \Longrightarrow \text{IDEM } (\mathcal{I}_R R)$  **unfolding** *IDEM-def Int-rel-def by smt*  
**lemma** *rI-Barcan*:  $\forall \pi. (\forall x. (\mathcal{I}_R R)(\pi x)) \preceq (\mathcal{I}_R R)(\forall x. \pi x)$  **unfolding** *Int-rel-def by simp*

A specialization relation can be derived from a given operation (intended as a closure-like operation).

**definition** *sp-rel*:: $(\sigma \Rightarrow \sigma) \Rightarrow (w \Rightarrow \sigma)$  ( $\mathcal{R}^C \mathcal{C}$ ) **where**  $\mathcal{R}^C \mathcal{C} \equiv \lambda w v. \mathcal{C} (\lambda u. u=v) w$

Preorder properties of the specialization relation follow directly from the corresponding operation's conditions.

**lemma** *sp-rel-reflex*:  $\text{EXP } \mathcal{C} \Longrightarrow \text{reflexive } (\mathcal{R}^C \mathcal{C})$  **by** (*simp add: EXP-def sp-rel-def*)  
**lemma** *sp-rel-trans*:  $\text{MONO } \mathcal{C} \Longrightarrow \text{IDEM } \mathcal{C} \Longrightarrow \text{transitive } (\mathcal{R}^C \mathcal{C})$  **by** (*smt IDEM-def MONO-def sp-rel-def*)

However, we can obtain finite countermodels for antisymmetry and symmetry given all relevant conditions. We will revisit this issue later and examine their relation with the topological separation axioms T0 and T1 resp.

**lemma** *iADDI C*  $\Longrightarrow \text{EXP } \mathcal{C} \Longrightarrow \text{NOR } \mathcal{C} \Longrightarrow \text{IDEM } \mathcal{C} \Longrightarrow \text{antisymmetric } (\mathcal{R}^C \mathcal{C})$  **nitpick oops**  
**lemma** *iADDI C*  $\Longrightarrow \text{EXP } \mathcal{C} \Longrightarrow \text{NOR } \mathcal{C} \Longrightarrow \text{IDEM } \mathcal{C} \Longrightarrow \text{symmetric } (\mathcal{R}^C \mathcal{C})$  **nitpick oops**

## 5.2 Alexandrov topology

As mentioned previously, Alexandrov closure (and by duality interior) operations correspond to specialization relations. It is worth mentioning that in Alexandrov topologies every point has a minimal/smallest neighborhood, namely the set of points related to it by the specialization (aka. accessibility) relation. Alexandrov spaces are thus also called 'finitely generated'. We examine below minimal conditions under which these relations obtain.

**lemma** *sp-rel-a*:  $\text{MONO } \mathcal{C} \Longrightarrow \forall A. (\mathcal{C}_R (\mathcal{R}^C \mathcal{C})) A \preceq \mathcal{C} A$  **by** (*smt Cl-rel-def MONO-def sp-rel-def*)  
**lemma** *sp-rel-b*:  $i\text{ADDI-a } \mathcal{C} \Longrightarrow \forall A. (\mathcal{C}_R (\mathcal{R}^C \mathcal{C})) A \succeq \mathcal{C} A$  **proof** –  
**assume** *iaddia*:  $i\text{ADDI-a } \mathcal{C}$

**{ fix** *A*

**let**  $?S = \lambda B. \sigma. \exists w. w. A w \wedge B = (\lambda u. u=w)$

**have**  $A \approx (\bigvee ?S)$  **using** *supremum-def by auto*

**hence**  $\mathcal{C}(A) \approx \mathcal{C}(\bigvee ?S)$  **by** (*smt eq-ext*)

**moreover have**  $\bigvee \text{Ra}[?S] \approx (\mathcal{C}_R (\mathcal{R}^C \mathcal{C})) A$  **by** (*smt Cl-rel-def Ra-restr-ex sp-rel-def*)

**moreover from** *iaddia* **have**  $\mathcal{C}(\bigvee ?S) \preceq \bigvee \text{Ra}[?S]$  **unfolding** *iADDI-a-def by simp*

**ultimately have**  $\mathcal{C} A \preceq (\mathcal{C}_R (\mathcal{R}^C \mathcal{C})) A$  **by** *simp*

**} thus** *?thesis by simp*

**qed**

**lemma** *sp-rel*:  $i\text{ADDI } \mathcal{C} \Longrightarrow \forall A. \mathcal{C} A \approx (\mathcal{C}_R (\mathcal{R}^C \mathcal{C})) A$  **by** (*metis MONO-iADDIb iADDI-a-def iADDI-b-def iADDI-def sp-rel-a sp-rel-b*)

It is instructive to expand the definitions in the above lemma:

**lemma**  $iADDI\ \mathcal{C} \implies \forall A. \forall w. (\mathcal{C}\ A)\ w \longleftrightarrow (\exists v. A\ v \wedge (\mathcal{C}\ (\lambda u. u=v))\ w)$  **using** *Cl-rel-def sp-rel* **by** *fastforce*

We now turn to the more traditional characterization of Alexandrov topologies in terms of closure under infinite joins/meets.

Fixed points of operations satisfying ADDI (MULT) are not in general closed under infinite joins (meets). For the given conditions countermodels are expected to be infinite. We (sanity) check that nitpick cannot find any.

**lemma**  $ADDI(\varphi) \implies \text{supremum-closed}\ (fp\ \varphi)$  **oops**

**lemma**  $MULT(\varphi) \implies \text{infimum-closed}\ (fp\ \varphi)$  **oops**

By contrast, we can show that this obtains if assuming the corresponding infinitary variants (iADDI/iMULT).

**lemma**  $iADDI(\varphi) \implies \text{supremum-closed}\ (fp\ \varphi)$  **by** (*metis (full-types) Ra-restr-ex iADDI-def supremum-def*)

**lemma**  $iMULT(\varphi) \implies \text{infimum-closed}\ (fp\ \varphi)$  **by** (*metis (full-types) Ra-restr-all iMULT-def infimum-def*)

As shown above, closure (interior) operations derived from relations readily satisfy iADDI (iMULT), being thus closed under infinite joins (meets).

**lemma**  $\text{supremum-closed}\ (fp\ (\mathcal{C}_R\ R))$  **by** (*smt Cl-rel-def supremum-def*)

**lemma**  $\text{infimum-closed}\ (fp\ (\mathcal{I}_R\ R))$  **by** (*smt Int-rel-def infimum-def*)

### 5.3 (Anti)symmetry and the separation axioms T0 and T1

We can now revisit the relationship between (anti)symmetry and the separation axioms T1 and T0.

T0: any two distinct points in the space can be separated by an open set (i.e. containing one point and not the other).

**abbreviation**  $T0\text{-sep}\ \mathcal{C} \equiv \forall w\ v. w \neq v \longrightarrow (\exists G. (fp\ \mathcal{C}^d)(G) \wedge (G\ w \neq G\ v))$

T1: any two distinct points can be separated by (two not necessarily disjoint) open sets, i.e. all singletons are closed.

**abbreviation**  $T1\text{-sep}\ \mathcal{C} \equiv \forall w. (fp\ \mathcal{C})(\lambda u. u = w)$

We can (sanity) check that T1 entails T0 but not viceversa.

**lemma**  $T0\text{-sep}\ \mathcal{C} \implies T1\text{-sep}\ \mathcal{C}$  **nitpick oops**

**lemma**  $T1\text{-sep}\ \mathcal{C} \implies T0\text{-sep}\ \mathcal{C}$  **by** (*smt compl-def dual-def dual-symm*)

Under appropriate conditions, T0-separation corresponds to antisymmetry of the specialization relation (here an ordering).

**lemma**  $T0\text{-sep}\ \mathcal{C} \longleftrightarrow \text{antisymmetric}\ (\mathcal{R}^{\mathcal{C}}\ \mathcal{C})$  **nitpick oops**

**lemma**  $T0\text{-antisymm-a}: \text{MONO}\ \mathcal{C} \implies T0\text{-sep}\ \mathcal{C} \longrightarrow \text{antisymmetric}\ (\mathcal{R}^{\mathcal{C}}\ \mathcal{C})$  **by** (*smt Cl-rel-def compl-def dual-def sp-rel-a*)

**lemma**  $T0\text{-antisymm-b}: \text{EXP}\ \mathcal{C} \implies \text{IDEM}\ \mathcal{C} \implies \text{antisymmetric}\ (\mathcal{R}^{\mathcal{C}}\ \mathcal{C}) \longrightarrow T0\text{-sep}\ \mathcal{C}$  **by** (*metis (full-types) EXP-dual1 IDEM-def IDEM-dual2 IDEMa-def IDEMb-def compl-def dEXP-def dual-def dual-symm sp-rel-def*)

**lemma**  $T0\text{-antisymm}: \text{MONO}\ \mathcal{C} \implies \text{EXP}\ \mathcal{C} \implies \text{IDEM}\ \mathcal{C} \implies T0\text{-sep}\ \mathcal{C} = \text{antisymmetric}\ (\mathcal{R}^{\mathcal{C}}\ \mathcal{C})$  **by** (*metis T0-antisymm-a T0-antisymm-b*)

Also, under the appropriate conditions, T1-separation corresponds to symmetry of the specialization relation.

**lemma** *T1-symm-a*:  $T1\text{-sep } \mathcal{C} \longrightarrow \text{symmetric } (\mathcal{R}^C \mathcal{C})$  **using** *sp-rel-def* **by** *auto*  
**lemma** *T1-symm-b*:  $MONO \mathcal{C} \Longrightarrow EXP \mathcal{C} \Longrightarrow T0\text{-sep } \mathcal{C} \Longrightarrow \text{symmetric } (\mathcal{R}^C \mathcal{C}) \longrightarrow T1\text{-sep } \mathcal{C}$  **by**  
*(metis T0-antisymm-a sp-rel-def sp-rel-reflex)*  
**lemma** *T1-symm*:  $MONO \mathcal{C} \Longrightarrow EXP \mathcal{C} \Longrightarrow T0\text{-sep } \mathcal{C} \Longrightarrow \text{symmetric } (\mathcal{R}^C \mathcal{C}) = T1\text{-sep } \mathcal{C}$  **by** *(metis T1-symm-a T1-symm-b)*

**end**

**theory** *topo-frontier-algebra*

**imports** *topo-operators-basic*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 6 Frontier Algebra

The closure of a set  $A$  ( $\mathcal{C}(A)$ ) can be seen as the set  $A$  augmented by (i) its boundary points, or (ii) its accumulation/limit points. In this section we explore the first variant by drawing on the notion of a frontier algebra, defined in an analogous fashion as the well-known closure and interior algebras.

Declares a primitive (unconstrained) frontier (aka. boundary) operation and defines others from it.

**consts**  $\mathcal{F}::\sigma\Rightarrow\sigma$

**abbreviation**  $\mathcal{I} \equiv \mathcal{I}_F \mathcal{F}$  — interior

**abbreviation**  $\mathcal{C} \equiv \mathcal{C}_F \mathcal{F}$  — closure

**abbreviation**  $\mathcal{B} \equiv \mathcal{B}_F \mathcal{F}$  — border

### 6.1 Basic properties

Verifies minimal conditions under which operators resulting from conversion functions coincide.

**lemma** *ICdual*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{I} \equiv \mathcal{C}^d$  **by** *(simp add: Cl-fr-def Fr-2-def Int-fr-def dual-def equal-op-def conn)*

**lemma** *ICdual'*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{C} \equiv \mathcal{I}^d$  **by** *(simp add: Cl-fr-def Fr-2-def Int-fr-def dual-def equal-op-def conn)*

**lemma** *BI-rel*:  $\mathcal{B} \equiv \mathcal{B}_I \mathcal{I}$  **using** *Br-fr-def Br-int-def Int-fr-def* **unfolding** *equal-op-def conn* **by** *auto*

**lemma** *IB-rel*:  $\mathcal{I} \equiv \mathcal{I}_B \mathcal{B}$  **using** *Br-fr-def Int-br-def Int-fr-def* **unfolding** *equal-op-def conn* **by** *auto*

**lemma** *BC-rel*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{B} \equiv \mathcal{B}_C \mathcal{C}$  **using** *BI-BC-rel BI-rel ICdual' eq-ext'* **by** *fastforce*

**lemma** *CB-rel*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{C} \equiv \mathcal{C}_B \mathcal{B}$  **using** *Br-fr-def Cl-br-def Cl-fr-def Fr-2-def* **unfolding** *equal-op-def conn* **by** *auto*

**lemma** *FI-rel*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{F} \equiv \mathcal{F}_I \mathcal{I}$  **by** *(smt Cl-fr-def Fr-int-def ICdual' Int-fr-def compl-def diff-def equal-op-def join-def meet-def)*

**lemma** *FC-rel*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{F} \equiv \mathcal{F}_C \mathcal{C}$  **by** *(metis (mono-tags, lifting) FI-rel Fr-2-def Fr-cl-def Fr-int-def ICdual' dual-def eq-ext' equal-op-def)*

**lemma** *FB-rel*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{F} \equiv \mathcal{F}_B \mathcal{B}$  **by** *(smt Br-fr-def CB-rel Cl-br-def FC-rel Fr-br-def Fr-cl-def equal-op-def join-def meet-def)*

Fixed-point and other operators are interestingly related.

**lemma** *fp1*:  $\mathcal{I}^{fp} \equiv \mathcal{B}^c$  **using** *Br-fr-def Int-fr-def* **unfolding** *equal-op-def conn* **by** *auto*

**lemma** *fp2*:  $\mathcal{B}^{fp} \equiv \mathcal{I}^c$  **using** *Br-fr-def Int-fr-def* **unfolding** *equal-op-def conn* **by** *auto*

**lemma** *fp3*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow \mathcal{C}^{fp} \equiv \mathcal{B}^d$  **using** *Br-fr-def Cl-fr-def Fr-2-def dual-def equal-op-def conn* **by** *fastforce*

**lemma** *fp4*:  $Fr\text{-}2 \mathcal{F} \Longrightarrow (\mathcal{B}^d)^{fp} \equiv \mathcal{C}$  **by** *(smt dimp-def equal-op-def fp3)*

**lemma** *fp5*:  $\mathcal{F}^{fp} \equiv \mathcal{B} \sqcup (\mathcal{C}^c)$  **using** *Br-fr-def Cl-fr-def* **unfolding** *equal-op-def conn* **by** *auto*

Different inter-relations (some redundant ones are kept to help the provers).

**lemma monI:**  $Fr-1b \mathcal{F} \implies MONO(\mathcal{I})$  by (simp add: IF1a MONO-MULTa)  
**lemma monC:**  $Fr-6 \mathcal{F} \implies MONO(\mathcal{C})$  by (simp add: Cl-fr-def Fr-6-def MONO-def conn)  
**lemma pB1:**  $\forall A. \mathcal{B} A \approx A \leftarrow \mathcal{I} A$  using Br-fr-def fp1 **unfolding conn equal-op-def by metis**  
**lemma pB2:**  $\forall A. \mathcal{B} A \approx A \wedge \mathcal{F} A$  by (simp add: Br-fr-def)  
**lemma pB3:**  $Fr-2 \mathcal{F} \implies \forall A. \mathcal{B}(-A) \approx -A \wedge \mathcal{F} A$  by (simp add: Fr-2-def pB2 conn)  
**lemma pB4:**  $Fr-2 \mathcal{F} \implies \forall A. \mathcal{B}(-A) \approx -A \wedge \mathcal{C} A$  using CB-rel Cl-br-def pB3 **unfolding conn equal-op-def by metis**  
**lemma pB5:**  $Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies \forall A. \mathcal{B}(C A) \preceq (\mathcal{B} A) \vee \mathcal{B}(-A)$  by (smt BC-rel Br-cl-def Cl-fr-def Fr-6-def PF6 equal-op-def conn)  
**lemma pF1:**  $\forall A. \mathcal{F} A \approx C A \leftarrow \mathcal{I} A$  using Cl-fr-def Int-fr-def conn by auto  
**lemma pF2:**  $Fr-2 \mathcal{F} \implies \forall A. \mathcal{F} A \approx C A \wedge \mathcal{C}(-A)$  using Cl-fr-def Fr-2-def conn by auto  
**lemma pF3:**  $Fr-2 \mathcal{F} \implies \forall A. \mathcal{F} A \approx \mathcal{B} A \vee \mathcal{B}(-A)$  using Br-fr-def Fr-2-def conn by auto  
**lemma pF4:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4(\mathcal{F}) \implies \forall A. \mathcal{F}(\mathcal{I} A) \preceq \mathcal{F} A$  by (smt IDEMa-def IF2 IF4 Int-fr-def MONO-def PF1 PF6 PI4 diff-def monC pF1)  
**lemma pF5:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall A. \mathcal{F}(C A) \preceq \mathcal{F} A$  by (metis Br-fr-def CF4 Cl-fr-def IDEM-def PF1 join-def meet-def pB5 pF3)  
**lemma pA1:**  $\forall A. A \approx \mathcal{I} A \vee \mathcal{B} A$  using Br-fr-def Int-fr-def **unfolding conn by auto**  
**lemma pA2:**  $Fr-2 \mathcal{F} \implies \forall A. A \approx C A \leftarrow \mathcal{B}(-A)$  using pB1 dual-def fp3 **unfolding equal-op-def conn by smt**  
**lemma pC1:**  $Fr-2 \mathcal{F} \implies \forall A. C A \approx A \vee \mathcal{B}(-A)$  using Cl-fr-def pB4 conn by auto  
**lemma pC2:**  $\forall A. C A \approx A \vee \mathcal{F} A$  using Cl-fr-def by auto  
**lemma pI1:**  $\forall A. \mathcal{I} A \approx A \leftarrow \mathcal{B} A$  using pA1 pB1 conn by auto  
**lemma pI2:**  $\forall A. \mathcal{I} A \approx A \leftarrow \mathcal{F} A$  by (simp add: Int-fr-def)

**lemma IC-imp:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall A B. \mathcal{I}(A \rightarrow B) \preceq C A \rightarrow C B$  **proof** –  
 assume fr1:  $Fr-1 \mathcal{F}$  and fr2:  $Fr-2 \mathcal{F}$  and fr3:  $Fr-3 \mathcal{F}$   
 { fix a b  
 have  $(a \rightarrow b) \wedge -b \rightarrow -a = \top$  **unfolding conn by auto**  
 hence  $\mathcal{I}((a \rightarrow b) \wedge -b \rightarrow -a) \approx \mathcal{I}(\top)$  by simp  
 hence  $\mathcal{I}((a \rightarrow b) \wedge -b \rightarrow -a) \approx \top$  using fr3 IF3 dNOR-def fr2 by auto  
 moreover have let  $A=(a \rightarrow b) \wedge -b; B=-a$  in  $\mathcal{I}(A \rightarrow B) \preceq \mathcal{I}(A) \rightarrow \mathcal{I}(B)$  using fr1 IF1 PI7 Int-7-def by metis  
 ultimately have  $\mathcal{I}((a \rightarrow b) \wedge -b) \rightarrow \mathcal{I}(-a) \approx \top$  **unfolding conn by simp**  
 moreover have let  $A=a \rightarrow b; B=-b$  in  $\mathcal{I}(A \wedge B) \approx \mathcal{I}(A) \wedge \mathcal{I}(B)$  using fr1 IF1 MULT-def by simp  
 ultimately have  $\mathcal{I}(a \rightarrow b) \wedge \mathcal{I}(-b) \rightarrow \mathcal{I}(-a) \approx \top$  **unfolding conn by simp**  
 moreover have  $\forall A. \mathcal{I}(-A) \approx -\mathcal{C}(A)$  using Cl-fr-def Fr-2-def Int-fr-def fr2 **unfolding conn by auto**  
 ultimately have  $\mathcal{I}(a \rightarrow b) \wedge -\mathcal{C}(b) \rightarrow -\mathcal{C}(a) \approx \top$  **unfolding conn by simp**  
 hence  $\mathcal{I}(a \rightarrow b) \wedge -\mathcal{C}(b) \preceq -\mathcal{C}(a)$  **unfolding conn by simp**  
 hence  $\mathcal{I}(a \rightarrow b) \wedge C a \preceq C b$  **unfolding conn by metis**  
 } thus ?thesis **unfolding conn by simp**  
**qed**

Defines some fixed-point predicates and prove some properties.

**abbreviation** *openset* (*Op*) **where**  $Op A \equiv fp \mathcal{I} A$

**abbreviation** *closedset* (*Cl*) **where**  $Cl A \equiv fp C A$

**abbreviation** *borderset* (*Br*) **where**  $Br A \equiv fp \mathcal{B} A$

**abbreviation** *frontierset* (*Fr*) **where**  $Fr A \equiv fp \mathcal{F} A$

**lemma Int-Open:**  $Fr-1a \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall A. Op(\mathcal{I} A)$  using IF4 IDEM-def by blast

**lemma Cl-Closed:**  $Fr-1a \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall A. Cl(C A)$  using CF4 IDEM-def by blast

**lemma Br-Border:**  $Fr-1b \mathcal{F} \implies \forall A. Br(\mathcal{B} A)$  using Br-fr-def Fr-1b-def conn by auto

In contrast, there is no analogous fixed-point result for frontier:



**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall A. Fr(\mathcal{F} A)$  **nitpick oops**

**lemma**  $OpClDual: Fr-2 \mathcal{F} \implies \forall A. Cl A \longleftrightarrow Op(-A)$  **using**  $Cl-fr-def Fr-2-def Int-fr-def conn$  **by auto**

**lemma**  $ClOpDual: Fr-2 \mathcal{F} \implies \forall A. Op A \longleftrightarrow Cl(-A)$  **using**  $ICdual dual-def unfolding equal-op-def conn$  **by auto**

**lemma**  $Fr-ClBr: \forall A. Fr(A) = (Cl(A) \wedge Br(A))$  **using**  $Br-fr-def Cl-fr-def join-def meet-def$  **by auto**

**lemma**  $Cl-F: Fr-4 \mathcal{F} \implies \forall A. Cl(\mathcal{F} A)$  **using**  $Cl-fr-def Fr-4-def conn$  **by auto**

## 6.2 Further properties

The definitions and theorems below are well known in the literature (e.g. [9]). Here we uncover the minimal conditions under which they hold (taking frontier operation as primitive).

**lemma**  $Cl-Bzero: Fr-2 \mathcal{F} \implies \forall A. Cl A \longleftrightarrow \mathcal{B}(-A) \approx \perp$  **using**  $pA2 pC1 unfolding conn$  **by metis**

**lemma**  $Op-Bzero: \forall A. Op A \longleftrightarrow (\mathcal{B} A) \approx \perp$  **using**  $pB1 pI1 unfolding conn$  **by metis**

**lemma**  $Br-boundary: Fr-2 \mathcal{F} \implies \forall A. Br(A) \longleftrightarrow \mathcal{I} A \approx \perp$  **by** ( $metis Br-fr-def Int-fr-def bottom-def diff-def meet-def$ )

**lemma**  $Fr-nowhereDense: Fr-2 \mathcal{F} \implies \forall A. Fr(A) \longrightarrow \mathcal{I}(C A) \approx \perp$  **using**  $Fr-ClBr Br-boundary eq-ext$  **by metis**

**lemma**  $Cl-FB: \forall A. Cl A \longleftrightarrow \mathcal{F} A \approx \mathcal{B} A$  **using**  $Br-fr-def Cl-fr-def unfolding conn$  **by auto**

**lemma**  $Op-FB: Fr-2 \mathcal{F} \implies \forall A. Op A \longleftrightarrow \mathcal{F} A \approx \mathcal{B}(-A)$  **using**  $Br-fr-def Fr-2-def Int-fr-def unfolding conn$  **by auto**

**lemma**  $Clopen-Fzero: \forall A. Cl A \wedge Op A \longleftrightarrow \mathcal{F} A \approx \perp$  **using**  $Cl-fr-def Int-fr-def unfolding conn$  **by auto**

**lemma**  $Int-sup-closed: Fr-1b \mathcal{F} \implies supremum-closed (\lambda A. Op A)$  **by** ( $smt IF1a Int-fr-def MONO-def MONO-MULTa sup-char conn$ )

**lemma**  $Int-meet-closed: Fr-1a \mathcal{F} \implies meet-closed (\lambda A. Op A)$  **using**  $Fr-1a-def Int-fr-def unfolding conn$  **by metis**

**lemma**  $Int-inf-closed: Fr-inf \mathcal{F} \implies infimum-closed (\lambda A. Op A)$  **by** ( $simp add: fp-IF-inf-closed$ )

**lemma**  $Cl-inf-closed: Fr-6 \mathcal{F} \implies infimum-closed (\lambda A. Cl A)$  **by** ( $smt Cl-fr-def Fr-6-def infimum-def join-def$ )

**lemma**  $Cl-join-closed: Fr-1a \mathcal{F} \implies Fr-2 \mathcal{F} \implies join-closed (\lambda A. Cl A)$  **using**  $ADDI-a-def CF1a CF2 EXP-def unfolding conn$  **by metis**

**lemma**  $Cl-sup-closed: Fr-2 \mathcal{F} \implies Fr-inf \mathcal{F} \implies supremum-closed (\lambda A. Cl A)$  **by** ( $simp add: fp-CF-sup-closed$ )

**lemma**  $Br-inf-closed: Fr-1b \mathcal{F} \implies infimum-closed (\lambda A. Br A)$  **by** ( $smt BI-rel Br-int-def IF1a MONO-iMULTa MONO-MULTa Ra-restr-all eq-ext' iMULT-a-def inf-char diff-def$ )

**lemma**  $Fr-inf-closed: Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies infimum-closed (\lambda A. Fr A)$  **by** ( $metis (full-types) Br-fr-def Br-inf-closed PF6 Cl-fr-def Cl-inf-closed meet-def join-def$ )

**lemma**  $Br-Fr-join: Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall A B. Br A \wedge Fr B \longrightarrow Br(A \vee B)$  **proof** –  
**assume**  $fr1: Fr-1 \mathcal{F}$  **and**  $fr2: Fr-2 \mathcal{F}$  **and**  $fr4: Fr-4 \mathcal{F}$

{ **fix**  $A B$

{ **assume**  $bra: Br A$  **and**  $frb: Fr B$

**from**  $bra$  **have**  $\mathcal{I} A \approx \perp$  **using**  $Br-boundary fr2$  **by auto**

**hence**  $1: \mathcal{C}(-A) \approx \top$  **by** ( $metis ICdual bottom-def compl-def dual-def eq-ext' fr2 top-def$ )

**from**  $frb$  **have**  $\mathcal{I}(C B) \approx \perp$  **by** ( $simp add: Fr-nowhereDense fr2$ )

**hence**  $2: \mathcal{C}(-(C B)) \approx \top$  **by** ( $metis ICdual bottom-def compl-def dual-def eq-ext' fr2 top-def$ )

**from**  $fr1 fr2$  **have**  $\mathcal{C}(-A) \leftarrow C B \preceq \mathcal{C}((-A) \leftarrow B)$  **using**  $CF1 Cl-6-def PC6$  **by metis**

**hence**  $\mathcal{C}(-A) \leftarrow C B \preceq \mathcal{C}(-(A \vee B))$  **unfolding conn** **by simp**

**hence**  $\top \leftarrow C B \preceq \mathcal{C}(-(A \vee B))$  **using**  $1$  **unfolding conn** **by simp**

**hence**  $3: -(C B) \preceq \mathcal{C}(-(A \vee B))$  **unfolding conn** **by simp**

**from**  $fr1 fr2 fr4$  **have**  $4: let M = -(C B); N = -(A \vee B) in M \preceq C N \longrightarrow C M \preceq C N$  **using**

$PC9 CF4 Cl-9-def PF1 CF1b$  **by fastforce**

**from**  $3 4$  **have**  $\mathcal{C}(-(C B)) \preceq \mathcal{C}(-(A \vee B))$  **by simp**

**hence**  $\top \approx \mathcal{C}(-(A \vee B))$  **using**  $2$  **unfolding top-def** **by simp**

**hence**  $\perp \approx \mathcal{I}(A \vee B)$  **using**  $fr2 ICdual dual-def eq-ext' conn$  **by metis**

hence  $Br (A \vee B)$  **using**  $fr2$  *Br-boundary* **by** *simp*  
 } hence  $Br A \wedge Fr B \longrightarrow Br (A \vee B)$  **by** *simp*  
 } hence  $\forall A B. Br A \wedge Fr B \longrightarrow Br (A \vee B)$  **by** *simp*  
 thus *?thesis* **by** *simp*

qed

**lemma** *Fr-join-closed*:  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow join-closed (\lambda A. Fr A)$  **by** (*smt Br-Fr-join ADDI-a-def CF1a Cl-fr-def PF1 diff-def join-def meet-def pB2 pF1*)

Introduces a predicate for indicating that two sets are disjoint and proves some properties.

**abbreviation**  $Disj A B \equiv A \wedge B \approx \perp$

**lemma** *Disj-comm*:  $\forall A B. Disj A B \longrightarrow Disj B A$  **unfolding** *conn* **by** *fastforce*

**lemma** *Disj-IF*:  $\forall A. Disj (\mathcal{I} A) (\mathcal{F} A)$  **by** (*simp add: Int-fr-def conn*)

**lemma** *Disj-B*:  $\forall A. Disj (\mathcal{B} A) (\mathcal{B}(-A))$  **using** *Br-fr-def* **unfolding** *conn* **by** *auto*

**lemma** *Disj-I*:  $\forall A. Disj (\mathcal{I} A) (-A)$  **by** (*simp add: Int-fr-def conn*)

**lemma** *Disj-BCI*:  $Fr-2 \mathcal{F} \Longrightarrow \forall A. Disj (\mathcal{B}(C A)) (\mathcal{I}(-A))$  **using** *Br-fr-def Cl-fr-def Fr-2-def Int-fr-def conn* **by** *metis*

**lemma** *Disj-CBI*:  $Fr-6 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow \forall A. Disj (C(\mathcal{B}(-A))) (\mathcal{I}(-A))$  **by** (*metis Br-fr-def Cl-F IB-rel Int-br-def MONO-MULTa MULT-a-def eq-ext' monC conn*)

Introduces a predicate for indicating that two sets are separated and proves some properties.

**definition**  $Sep A B \equiv Disj (C A) B \wedge Disj (C B) A$

**lemma** *Sep-comm*:  $\forall A B. Sep A B \longrightarrow Sep B A$  **by** (*simp add: Sep-def*)

**lemma** *Sep-disj*:  $\forall A B. Sep A B \longrightarrow Disj A B$  **using** *Cl-fr-def Sep-def conn* **by** *fastforce*

**lemma** *Sep-I*:  $Fr-1(\mathcal{F}) \Longrightarrow Fr-2(\mathcal{F}) \Longrightarrow Fr-4(\mathcal{F}) \Longrightarrow \forall A. Sep (\mathcal{I} A) (\mathcal{I}(-A))$  **using** *Cl-fr-def pF4 Fr-2-def Int-fr-def* **unfolding** *Sep-def conn* **by** *metis*

**lemma** *Sep-sub*:  $Fr-6 \mathcal{F} \Longrightarrow \forall A B C D. Sep A B \wedge C \preceq A \wedge D \preceq B \longrightarrow Sep C D$  **using** *MONO-def monC* **unfolding** *Sep-def conn* **by** *smt*

**lemma** *Sep-Cl-diff*:  $Fr-6 \mathcal{F} \Longrightarrow \forall A B. Cl(A) \wedge Cl(B) \longrightarrow Sep (A \leftarrow B) (B \leftarrow A)$  **using** *Fr-6-def pC2* **unfolding** *Sep-def conn* **by** *smt*

**lemma** *Sep-Op-diff*:  $Fr-1b \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow \forall A B. Op(A) \wedge Op(B) \longrightarrow Sep (A \leftarrow B) (B \leftarrow A)$   
**proof** –

**assume**  $fr1b:Fr-1b \mathcal{F}$  **and**  $fr2:Fr-2 \mathcal{F}$

  { **fix**  $A B$

**from**  $fr1b$   $fr2$  **have**  $aux: let M=-A ; N=-B in (Cl(M) \wedge Cl(N) \longrightarrow Sep (M \leftarrow N) (N \leftarrow M))$

**using** *PF6 Sep-Cl-diff* **by** *simp*

  { **assume**  $Op(A) \wedge Op(B)$

**hence**  $Cl(-A) \wedge Cl(-B)$  **using**  $fr2$  *ClOpdual* **by** *simp*

**hence**  $Sep (-A \leftarrow -B) (-B \leftarrow -A)$  **using**  $fr1b$   $fr2$   $aux$  **unfolding** *conn* **by** *simp*

**moreover** **have**  $(-A \leftarrow -B) \approx (B \leftarrow A)$  **unfolding** *conn* **by** *auto*

**moreover** **have**  $(-B \leftarrow -A) \approx (A \leftarrow B)$  **unfolding** *conn* **by** *auto*

**ultimately** **have**  $Sep (B \leftarrow A) (A \leftarrow B)$  **unfolding** *conn* **by** *simp*

**hence**  $Sep (A \leftarrow B) (B \leftarrow A)$  **using** *Sep-comm* **by** *simp*

  } **hence**  $Op(A) \wedge Op(B) \longrightarrow Sep (A \leftarrow B) (B \leftarrow A)$  **by** (*rule impI*)

  } **thus** *?thesis* **by** *simp*

qed

**lemma** *Sep-Cl*:  $\forall A B. Cl(A) \wedge Cl(B) \wedge Disj A B \longrightarrow Sep A B$  **unfolding** *Sep-def conn* **by** *blast*

**lemma** *Sep-Op*:  $Fr-1b \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow \forall A B. Op(A) \wedge Op(B) \wedge Disj A B \longrightarrow Sep A B$  **proof** –

**assume**  $fr1b:Fr-1b \mathcal{F}$  **and**  $fr2:Fr-2 \mathcal{F}$

  { **fix**  $A B$

**from**  $fr1b$   $fr2$  **have**  $aux: Op(A) \wedge Op(B) \longrightarrow Sep (A \leftarrow B) (B \leftarrow A)$  **using** *Sep-Op-diff* **by** *simp*

  { **assume**  $op: Op(A) \wedge Op(B)$  **and**  $disj: Disj A B$

**hence**  $(A \leftarrow B) \approx A \wedge (B \leftarrow A) \approx B$  **unfolding** *conn* **by** *blast*

**hence**  $Sep A B$  **using**  $op$   $aux$  **unfolding** *conn* **by** *simp*

} hence  $Op(A) \wedge Op(B) \wedge Disj A B \longrightarrow Sep A B$  by *simp*  
 } thus *?thesis* by *simp*  
**qed**  
**lemma** *Fr-1a*  $\mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow \forall A B C. Sep A B \wedge Sep A C \longrightarrow Sep A (B \vee C)$  **using** *CF1a ADDI-a-def unfolding Sep-def conn by metis*

Verifies a neighborhood-based definition of closure.

**definition** *nbhd*  $A p \equiv \exists E. E \preceq A \wedge Op(E) \wedge (E p)$   
**lemma** *nbhd-def2*:  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow \forall A p. (nbhd A p) = (\mathcal{I} A p)$  **using** *pF4 MONO-def PF1 monI pI2 unfolding nbhd-def conn by smt*

**lemma** *C-def-lr*:  $Fr-1b \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow \forall A p. (C A p) \longrightarrow (\forall E. (nbhd E p) \longrightarrow \neg(Disj E A))$  **using** *Cl-fr-def Fr-2-def Fr-6-def PF6 pF1 unfolding nbhd-def conn by smt*

**lemma** *C-def-rl*:  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow \forall A p. (C A p) \longleftarrow (\forall E. (nbhd E p) \longrightarrow \neg(Disj E A))$  **using** *Cl-fr-def pF5 pA1 pB4 unfolding nbhd-def conn by smt*

**lemma** *C-def*:  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-4 \mathcal{F} \Longrightarrow \forall A p. (C A p) \longleftrightarrow (\forall E. (nbhd E p) \longrightarrow \neg(Disj E A))$  **using** *C-def-lr C-def-rl PF1 by blast*

Explore the Barcan and converse Barcan formulas.

**lemma** *Barcan-I*:  $Fr-inf \mathcal{F} \Longrightarrow \forall P. (\forall x. \mathcal{I}(P x)) \preceq \mathcal{I}(\forall x. P x)$  **using** *IF-inf Barcan1 by auto*

**lemma** *Barcan-C*:  $Fr-2 \mathcal{F} \Longrightarrow Fr-inf \mathcal{F} \Longrightarrow \forall P. \mathcal{C}(\exists x. P x) \preceq (\exists x. \mathcal{C}(P x))$  **using** *Fr-2-def CF-inf Barcan2 by metis*

**lemma** *CBarcan-I*:  $Fr-1b \mathcal{F} \Longrightarrow \forall P. \mathcal{I}(\forall x. P x) \preceq (\forall x. \mathcal{I}(P x))$  **by** (*metis (mono-tags, lifting) MONO-def monI*)

**lemma** *CBarcan-C*:  $Fr-6 \mathcal{F} \Longrightarrow \forall P. (\exists x. \mathcal{C}(P x)) \preceq \mathcal{C}(\exists x. P x)$  **by** (*metis (mono-tags, lifting) MONO-def monC*)

**end**

**theory** *topo-negation-conditions*

**imports** *topo-frontier-algebra sse-operation-negative-quantification*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 7 Frontier-based negation - Semantic conditions

We define a paracomplete and a paraconsistent negation employing the interior and closure operation resp. We take the frontier operator as primitive and explore which semantic conditions are minimally required to obtain some relevant properties of negation.

**definition** *neg-I*:  $\sigma \Rightarrow \sigma (\neg^I)$  **where**  $\neg^I A \equiv \mathcal{I}(\neg A)$

**definition** *neg-C*:  $\sigma \Rightarrow \sigma (\neg^C)$  **where**  $\neg^C A \equiv \mathcal{C}(\neg A)$

**declare** *neg-I-def*[*conn*] *neg-C-def*[*conn*]

(We rename the meta-logical HOL negation to avoid terminological confusion)

**abbreviation** *cneg*:  $bool \Rightarrow bool (\sim [40]40)$  **where**  $\sim \varphi \equiv \neg \varphi$

### 7.1 'Explosion' (ECQ), non-contradiction (LNC) and excluded middle (TND)

TND

**lemma**  $\mathfrak{F} \mathcal{F} \Longrightarrow TNDm \neg^I$  **nitpick oops**

**lemma** *TND-C*:  $TND \neg^C$  **by** (*simp add: pC2 Defs conn*)

ECQ

**lemma** *ECQ-I*:  $ECQ \neg^I$  **by** (*simp add: pI2 Defs conn*)

**lemma**  $\mathfrak{F} \mathcal{F} \implies ECQm \neg^C$  **nitpick oops**

LNC

**lemma** *LNC*  $\neg^I$  **nitpick oops**

**lemma** *LNC-I*:  $Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies LNC \neg^I$  **using** *ECQ-I ECQ-def IF3 LNC-def dNOR-def unfolding conn by auto*

**lemma** *LNC*  $\neg^C$  **nitpick oops**

**lemma** *LNC-C*:  $Fr-6 \mathcal{F} \implies LNC \neg^C$  **unfolding** *Defs* **by** (*smt Cl-fr-def MONO-def compl-def join-def meet-def monC neg-C-def top-def*)

Relations between LNC and different ECQ variants (only relevant for paraconsistent negation).

**lemma**  $ECQ \neg^C \longrightarrow LNC \neg^C$  **by** (*simp add: pC2 Defs conn*)

**lemma** *ECQw-LNC*:  $ECQw \neg^C \longrightarrow LNC \neg^C$  **by** (*smt ECQw-def LNC-def pC2 conn*)

**lemma** *ECQm-LNC*:  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies ECQm \neg^C \longrightarrow LNC \neg^C$  **using** *Cl-fr-def Fr-1-def pF2 unfolding Defs conn by metis*

**lemma**  $\mathfrak{F} \mathcal{F} \implies LNC \neg^C \longrightarrow ECQm \neg^C$  **nitpick oops**

Below we show that enforcing all conditions on the frontier operator still leaves room for both boldly paraconsistent and paracomplete models. We use Nitpick to generate a non-trivial model (here a set algebra with 8 elements).

**lemma**  $\mathfrak{F} \mathcal{F} \wedge \sim ECQm \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\mathfrak{F} \mathcal{F} \wedge \sim TNDm \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

## 7.2 Modus tollens (MT)

MT-I

**lemma** *MT0-I*:  $Fr-1b \mathcal{F} \implies MT0 \neg^I$  **unfolding** *Defs* **by** (*smt MONO-def compl-def monI neg-I-def top-def*)

**lemma** *MT1-I*:  $Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies MT1 \neg^I$  **unfolding** *Defs* **by** (*metis MONO-def monI IF3 Int-fr-def compl-def dNOR-def diff-def neg-I-def top-def*)

**lemma**  $\mathfrak{F} \mathcal{F} \implies MT2 \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge MT2 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1a \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge MT2 \neg^I$  **nitpick**[*satisfy*] **oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge MT2 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge MT2 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies MT3 \neg^I$  **nitpick oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1a \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge MT3 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\sim TNDm \neg^I \wedge MT0 \neg^I \wedge MT1 \neg^I \wedge MT2 \neg^I \wedge MT3 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

MT-C

**lemma**  $Fr-2 \mathcal{F} \implies MT0 \neg^C$  **nitpick oops**

**lemma** *MT0-C*:  $Fr-6 \mathcal{F} \implies MT0 \neg^C$  **unfolding** *Defs* **by** (*smt ICdual MONO-def compl-def monC neg-C-def top-def*)

**lemma** *MT1-C*:  $Fr-6 \mathcal{F} \implies MT1 \neg^C$  **unfolding** *Defs* **by** (*smt Cl-fr-def Fr-6-def conn*)

**lemma**  $\mathfrak{F} \mathcal{F} \implies MT2 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQm \neg^C \wedge \mathfrak{F} \mathcal{F} \wedge MT2 \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma** *MT3-C*:  $Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies MT3 \neg^C$  **unfolding** *Defs* **using** *MONO-def monI* **by** (*metis ClOpdual IF3 compl-def dNOR-def diff-def neg-C-def pA2 top-def*)

**lemma**  $\sim ECQm \neg^C \wedge MT0 \neg^C \wedge MT1 \neg^C \wedge MT2 \neg^C \wedge MT3 \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

### 7.3 Contraposition rules (CoP)

CoPw-I

**lemma**  $CoPw \neg^I$  **nitpick oops**

**lemma**  $CoPw-I: Fr-1b \mathcal{F} \implies CoPw \neg^I$  **unfolding** *Defs conn by (metis (no-types, lifting) MONO-def monI)*

CoPw-C

**lemma**  $CoPw \neg^C$  **nitpick oops**

**lemma**  $CoPw-C: Fr-6 \mathcal{F} \implies CoPw \neg^C$  **by** *(smt CoPw-def MONO-def monC conn)*

We can indeed prove that XCoP is entailed by CoP1 (CoP2) in the particular case of a closure-(interior-)based negation.

**lemma**  $CoP1-XCoP: CoP1 \neg^C \longrightarrow XCoP \neg^C$  **by** *(metis XCoP-def2 CoP1-def CoP1-def2 DM2-CoPw DM2-def ECQw-def TND-C TND-def TNDw-def top-def)*

**lemma**  $CoP2-XCoP: CoP2 \neg^I \longrightarrow XCoP \neg^I$  **by** *(smt XCoP-def2 CoP2-DM3 CoP2-def2 CoPw-def DM3-def DNE-def ECQ-I ECQ-def ECQw-def TNDw-def bottom-def join-def)*

CoP1-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP1 \neg^I$  **nitpick oops**

**lemma**  $\sim TNDm \neg^I \wedge \mathfrak{F} \mathcal{F} \wedge CoP1 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

CoP1-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP1 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge CoP1 \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $CoP1 \neg^C \longrightarrow ECQm \neg^C$  **using** *XCoP-def2 CoP1-XCoP ECQm-def ECQw-def* **by** *blast*

CoP2-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP2 \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge CoP2 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge CoP2 \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $CoP2 \neg^I \longrightarrow TNDm \neg^I$  **using** *XCoP-def2 CoP2-XCoP TNDm-def TNDw-def* **by** *auto*

CoP2-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP2 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQm \neg^C \wedge \mathfrak{F} \mathcal{F} \wedge CoP2 \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

CoP3-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP3 \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge CoP3 \neg^I$  **oops**

CoP3-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP3 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge CoP3 \neg^C$  **oops**

XCoP-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies XCoP \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge XCoP \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge XCoP \neg^I$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $XCoP \neg^I \longrightarrow TNDm \neg^I$  **by** *(simp add: XCoP-def2 TNDm-def TNDw-def)*

XCoP-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies XCoP \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge XCoP \neg^C$  **nitpick**[*satisfy, card w=3*] **oops**

**lemma**  $XCoP \neg^C \longrightarrow ECQm \neg^C$  **by** *(simp add: XCoP-def2 ECQm-def ECQw-def)*

## 7.4 Normality (negative) and its dual (nNor/nDNor)

nNor-I

**lemma**  $nNor \neg^I$  **nitpick oops**

**lemma**  $nNor-I: Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies nNor \neg^I$  **using** *IF3 dNOR-def unfolding Defs conn by auto*

nNor-C

**lemma**  $nNor-C: nNor \neg^C$  **unfolding** *Cl-fr-def Defs conn by simp*

nDNor-I

**lemma**  $nDNor-I: nDNor \neg^I$  **unfolding** *Int-fr-def Defs conn by simp*

nDNor-C

**lemma**  $nDNor \neg^C$  **nitpick oops**

**lemma**  $nDNor-C: Fr-3 \mathcal{F} \implies nDNor \neg^C$  **using** *pC2 NOR-def unfolding Defs conn by simp*

## 7.5 Double negation introduction/elimination (DNI/DNE)

DNI-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNI \neg^I$  **nitpick oops**

**lemma**  $\sim TNDm \neg^I \wedge \mathfrak{F} \mathcal{F} \wedge DNI \neg^I$  **nitpick[satisfy, card w=3] oops**

DNI-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNI \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNI \neg^C$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim ECQm \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNI \neg^C$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim ECQm \neg^C \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNI \neg^C$  **nitpick[satisfy, card w=3] oops**

DNE-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNE \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNE \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNE \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DNE \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TND \neg^I \wedge DNE \neg^I \wedge DNI \neg^I$  **oops**

DNE-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNE \neg^C$  **nitpick oops**

**lemma**  $\sim ECQm \neg^C \wedge \mathfrak{F} \mathcal{F} \wedge DNE \neg^C$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim ECQ \neg^C \wedge DNE \neg^C \wedge DNI \neg^C$  **oops**

rDNI-I

**lemma**  $Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies rDNI \neg^I$  **using** *nNor-I nDNor-I nDNor-rDNI by simp*

rDNI-C

**lemma**  $Fr-3 \mathcal{F} \implies rDNI \neg^C$  **using** *nNor-C nDNor-C nDNor-rDNI by simp*

rDNE-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies rDNE \neg^I$  **nitpick oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge rDNE \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge rDNE \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TNDm \neg^I \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge rDNE \neg^I$  **nitpick[satisfy, card w=3] oops**

rDNE-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies rDNE \neg^C$  **nitpick oops**

**lemma**  $\sim ECQm \neg^C \wedge \mathfrak{F} \mathcal{F} \wedge rDNE \neg^C$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim ECQm \neg^C \wedge rDNE \neg^C \wedge rDNI \neg^C$  **nitpick[satisfy, card w=3] oops**

## 7.6 De Morgan laws

DM1/2 (see CoPw)

DM3-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies DM3 \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DM3 \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DM3 \neg^I$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim TNDm \neg^I \wedge DM3 \neg^I$  **oops**

DM3-C

**lemma**  $DM3 \neg^C$  **nitpick oops**

**lemma**  $DM3-C: Fr-1a \mathcal{F} \implies Fr-2 \mathcal{F} \implies DM3 \neg^C$  **using**  $DM3-def Fr-1a-def pA2 pF2$  **unfolding conn by smt**

**lemma**  $\sim ECQm \neg^C \wedge \mathfrak{F} \mathcal{F} \wedge DM3 \neg^C$  **nitpick[satisfy, card w=3] oops**

DM4-I

**lemma**  $DM4 \neg^I$  **nitpick oops**

**lemma**  $DM4-I: Fr-1a \mathcal{F} \implies DM4 \neg^I$  **using**  $ADDI-a-def Cl-fr-def DM4-def IC1b IF1b dual-def$  **unfolding conn by smt**

**lemma**  $\sim TNDm \neg^I \wedge \mathfrak{F} \mathcal{F} \wedge DM4 \neg^I$  **nitpick[satisfy, card w=3] oops**

DM4-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies DM4 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge DM4 \neg^C$  **nitpick[satisfy, card w=3] oops**

**lemma**  $\sim ECQm \neg^C \wedge DM4 \neg^C$  **oops**

iDM1/2 (see CoPw)

iDM3-I

**lemma**  $\mathfrak{F} \mathcal{F} \implies Fr-inf \mathcal{F} \implies iDM3 \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge iDM3 \neg^I$  **nitpick[satisfy] oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge iDM3 \neg^I$  **nitpick[satisfy] oops**

**lemma**  $\sim TNDm \neg^I \wedge iDM3 \neg^I$  **oops**

iDM3-C

**lemma**  $iDM3 \neg^C$  **nitpick oops**

**lemma**  $iDM3-C: Fr-2 \mathcal{F} \implies Fr-inf \mathcal{F} \implies iDM3 \neg^C$  **unfolding**  $Defs$  **by**  $(metis (full-types) CF-inf Ra-restr-ex dom-compl-def iADDI-a-def iDM-a neg-C-def)$

iDM4-I

**lemma**  $iDM4 \neg^I$  **nitpick oops**

**lemma**  $iDM4-I: Fr-inf \mathcal{F} \implies iDM4 \neg^I$  **unfolding**  $Defs$  **by**  $(metis IF-inf Ra-restr-all dom-compl-def iDM-b iMULT-b-def neg-I-def)$

iDM4-C

**lemma**  $\mathfrak{F} \mathcal{F} \implies Fr-inf \mathcal{F} \implies iDM4 \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge iDM4 \neg^C$  **nitpick[satisfy] oops**

**lemma**  $\sim ECQm \neg^C \wedge iDM4 \neg^C$  **oops**

## 7.7 Local contraposition axioms (lCoP)

lCoPw-I

**lemma**  $\wp \mathcal{F} \implies lCoPw(\rightarrow) \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoPw(\rightarrow) \neg^I$  **nitpick[satisfy,card w=3] oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoPw(\rightarrow) \neg^I$  **nitpick[satisfy,card w=3] oops**

**lemma**  $lCoPw(\rightarrow) \neg^I \longrightarrow TNDm \neg^I$  **by** (*simp add: XCoP-def2 TNDm-def TNDw-def lCoPw-XCoP*)

lCoPw-C

**lemma**  $\wp \mathcal{F} \implies lCoPw(\rightarrow) \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoPw(\rightarrow) \neg^C$  **nitpick[satisfy,card w=3] oops**

**lemma**  $lCoPw(\rightarrow) \neg^C \longrightarrow ECQm \neg^C$  **by** (*simp add: XCoP-def2 ECQm-def ECQw-def lCoPw-XCoP*)

lCoP1-I

**lemma**  $\wp \mathcal{F} \implies lCoP1(\rightarrow) \neg^I$  **nitpick oops**

**lemma**  $lCoP1(\rightarrow) \neg^I \longrightarrow TND \neg^I$  **by** (*simp add: lCoP1-TND*)

lCoP1-C

**lemma**  $\wp \mathcal{F} \implies lCoP1(\rightarrow) \neg^C$  **nitpick oops**

**lemma**  $\sim ECQ \neg^C \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoP1(\rightarrow) \neg^C$  **nitpick[satisfy,card w=3] oops**

**lemma**  $lCoP1(\rightarrow) \neg^C \longrightarrow ECQm \neg^C$  **by** (*simp add: XCoP-def2 ECQm-def ECQw-def lCoP1-def2 lCoPw-XCoP*)

lCoP2-I

**lemma**  $\wp \mathcal{F} \implies lCoP2(\rightarrow) \neg^I$  **nitpick oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoP2(\rightarrow) \neg^I$  **nitpick[satisfy,card w=3] oops**

**lemma**  $\sim TND \neg^I \wedge Fr-1 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge lCoP2(\rightarrow) \neg^I$  **nitpick[satisfy,card w=3] oops**

**lemma**  $lCoP2(\rightarrow) \neg^I \longrightarrow TNDm \neg^I$  **by** (*simp add: XCoP-def2 TNDm-def TNDw-def lCoP2-def2 lCoPw-XCoP*)

lCoP2-C

**lemma**  $\wp \mathcal{F} \implies lCoP2(\rightarrow) \neg^C$  **nitpick oops**

**lemma**  $lCoP2(\rightarrow) \neg^C \longrightarrow ECQ \neg^C$  **by** (*simp add: lCoP2-ECQ*)

lCoP3-I

**lemma**  $\wp \mathcal{F} \implies lCoP3(\rightarrow) \neg^I$  **nitpick oops**

**lemma**  $lCoP3(\rightarrow) \neg^I \longrightarrow TND \neg^I$  **unfolding Defs conn by metis**

lCoP3-C

**lemma**  $\wp \mathcal{F} \implies lCoP3(\rightarrow) \neg^C$  **nitpick oops**

**lemma**  $lCoP3(\rightarrow) \neg^C \longrightarrow ECQ \neg^C$  **unfolding Defs conn by metis**

## 7.8 Disjunctive syllogism

DS1-I

**lemma**  $DS1(\rightarrow) \neg^I$  **using DS1-def ECQ-I ECQ-def unfolding conn by auto**

DS1-C

**lemma**  $\wp \mathcal{F} \implies DS1(\rightarrow) \neg^C$  **nitpick oops**

**lemma**  $DS1(\rightarrow) \neg^C \longrightarrow ECQ \neg^C$  **unfolding Defs conn by metis**

DS2-I



**lemma**  $\mathfrak{F} \mathcal{F} \implies DS2(\rightarrow) \neg^I$  **nitpick oops**  
**lemma**  $DS2(\rightarrow) \neg^I \longrightarrow TND \neg^I$  **by** (*simp add: Defs conn*)

DS2-C

**lemma**  $DS2(\rightarrow) \neg^C$  **using** *TND-C unfolding Defs conn by auto*

**end**

**theory** *topo-negation-fixedpoints*

**imports** *topo-frontier-algebra sse-operation-negative-quantification*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 8 Frontier-based negation - Fixed-points

We define a paracomplete and a paraconsistent negation employing the interior and the closure operation respectively. We explore the use of fixed-point predicates to recover some relevant properties of negation, many of which cannot be readily recovered by just adding semantic conditions. We take the frontier operator as primitive and explore which semantic conditions are minimally required.

**definition**  $neg-I::\sigma \Rightarrow \sigma \neg^I$  **where**  $\neg^I \varphi \equiv \mathcal{I}(-\varphi)$

**definition**  $neg-C::\sigma \Rightarrow \sigma \neg^C$  **where**  $\neg^C \varphi \equiv \mathcal{C}(-\varphi)$

**declare**  $neg-I-def[conn]$   $neg-C-def[conn]$

Note that all results obtained for fixed-point predicates extend to their associated operators as follows:

**lemma**  $\forall A. \gamma^{fp}(A) \wedge \varphi(A) \preceq \psi(A) \implies \forall A. (fp \ \gamma)(A) \longrightarrow \varphi(A) \preceq \psi(A)$  **unfolding** *conn by simp*

**lemma**  $\forall A \ B. \gamma^{fp}(A) \wedge \gamma^{fp}(B) \wedge (\varphi \ A \ B) \preceq (\psi \ A \ B) \implies \forall A \ B. (fp \ \gamma)(A) \wedge (fp \ \gamma)(B) \longrightarrow (\varphi \ A \ B) \preceq (\psi \ A \ B)$  **unfolding** *conn by simp*

Recall from previous results that if we have  $Fr(A)$  then we also have both  $Cl(A)$  and  $Br(A)$ . With this understanding we will tacitly assume the corresponding results for  $Fr(-)$  below. Moreover, we obtained countermodels (using Nitpick) for all formulas featuring other combinations (not shown).

### 8.1 'Explosion' (ECQ) and excluded middle (TND)

TND-I

**lemma**  $Fr-2 \ \mathcal{F} \implies \forall A. Cl(A) \longrightarrow TND^A \neg^I$  **by** (*simp add: OpCldual conn*)

ECQ-C

**lemma**  $Fr-2 \ \mathcal{F} \implies \forall A. Op(A) \longrightarrow ECQ^A \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

### 8.2 Contraposition rules

CoPw-I

**lemma**  $\forall A \ B. Br(-B) \longrightarrow CoPw^{AB} \neg^I$  **using** *Int-fr-def pB1 conn by auto*

**lemma**  $Fr-2 \ \mathcal{F} \implies \forall A \ B. Cl(A) \longrightarrow CoPw^{AB} \neg^I$  **using** *Int-fr-def OpCldual conn by auto*

CoPw-C

**lemma**  $Fr-2 \ \mathcal{F} \implies \forall A \ B. Br(A) \longrightarrow CoPw^{AB} \neg^C$  **using** *pA2 pB2 pF2 unfolding conn bymetis*

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow CoPw^{AB} \neg^C$  **using** *ClOpdual Cl-fr-def unfolding conn by auto*

CoP1-I

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Cl(A) \longrightarrow CoP1^{AB} \neg^I$  **using** *Int-fr-def OpClDual conn by auto*

**lemma** *Fr-1b*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow CoP1^{AB} \neg^I$  **by** (*smt IF2 dEXP-def MONO-def monI conn*)

**lemma** *CoP1-I-rec: Fr-2*  $\mathcal{F} \Longrightarrow$  *Fr-3*  $\mathcal{F} \Longrightarrow \forall A B. Br(-B) \longrightarrow CoP1^{AB} \neg^I$  **using** *IF3 dNOR-def Br-boundary unfolding conn by auto*

CoP1-C

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow CoP1^{AB} \neg^C$  **using** *Int-fr-def pC2 pF2 unfolding conn by metis*

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Br(A) \longrightarrow CoP1^{AB} \neg^C$  **using** *Br-fr-def Cl-fr-def pF2 unfolding conn by fastforce*

CoP2-I

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Cl(A) \longrightarrow CoP2^{AB} \neg^I$  **using** *Int-fr-def OpClDual unfolding conn by auto*

**lemma**  $\forall A B. Br(-B) \longrightarrow CoP2^{AB} \neg^I$  **by** (*simp add: pI1 conn*)

CoP2-C

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow CoP2^{AB} \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

**lemma** *Fr-6*  $\mathcal{F} \Longrightarrow \forall A B. Cl(A) \longrightarrow CoP2^{AB} \neg^C$  **by** (*smt Cl-fr-def MONO-def monC conn*)

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow$  *Fr-3*  $\mathcal{F} \Longrightarrow \forall A B. Br(A) \longrightarrow CoP2^{AB} \neg^C$  **using** *CoP1-I-rec Disj-IF pA2 pF2 pF3 unfolding conn by smt*

CoP3-I

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Cl(A) \longrightarrow CoP3^{AB} \neg^I$  **by** (*metis Disj-I ICdual' compl-def dual-def eq-ext' meet-def neg-I-def*)

CoP3-C

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow CoP3^{AB} \neg^C$  **by** (*metis Disj-I ICdual compl-def dual-def eq-ext' meet-def neg-C-def*)

XCoP-I

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Cl(A) \longrightarrow XCoP^{AB} \neg^I$  **using** *Fr-2-def pA1 pA2 pF1 unfolding conn by metis*

**lemma**  $\forall A B. Br(-B) \longrightarrow XCoP^{AB} \neg^I$  **using** *IB-rel Int-br-def diff-def eq-ext' conn by fastforce*

XCoP-C

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. Op(B) \longrightarrow XCoP^{AB} \neg^C$  **by** (*metis ClOpdual compl-def diff-def meet-def neg-C-def pA2*)

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow \forall A B. \forall A B. Br(A) \longrightarrow XCoP^{AB} \neg^C$  **using** *Cl-fr-def compl-def join-def neg-C-def pF3 by fastforce*

### 8.3 Double negation introduction/elimination

DNI-I

**lemma** *Fr-1b*  $\mathcal{F} \Longrightarrow \forall A. Op(A) \longrightarrow DNI^A \neg^I$  **using** *MONO-def monI pA1 unfolding conn by smt*

**lemma** *Fr-2*  $\mathcal{F} \Longrightarrow$  *Fr-3*  $\mathcal{F} \Longrightarrow \forall A. Br(-A) \longrightarrow DNI^A \neg^I$  **using** *CoP1-I-rec by simp*

DNI-C

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A. Op(A) \longrightarrow DNI^A \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

DNE-I

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A. Cl(A) \longrightarrow DNE^A \neg^I$  **by** (*simp add: Cl-fr-def Fr-2-def Int-fr-def conn*)

DNE-C

**lemma** *Fr-6*  $\mathcal{F} \implies \forall A. Cl(A) \longrightarrow DNE^A \neg^C$  **by** (*smt MONO-def monC pC2 conn*)

**lemma** *Fr-2*  $\mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall A. Br(A) \longrightarrow DNE^A \neg^C$  **using** *CoP1-I-rec Disj-IF pA2 pF2 pF3 unfolding conn by smt*

## 8.4 De Morgan laws

DM1-I

**lemma** *Fr-1b*  $\mathcal{F} \implies \forall A B. DM1^{AB} \neg^I$  **by** (*smt MONO-def monI conn*)

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Cl(A) \wedge Cl(B) \longrightarrow DM1^{AB} \neg^I$  **using** *pF2 pI2 conn by fastforce*

DM1-C

**lemma** *Fr-6*  $\mathcal{F} \implies \forall A B. DM1^{AB} \neg^C$  **by** (*smt MONO-def monC conn*)

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Br(A) \wedge Br(B) \longrightarrow DM1^{AB} \neg^C$  **using** *CF2 EXP-def pF2 pF3 unfolding conn by metis*

DM2-I

**lemma** *Fr-1b*  $\mathcal{F} \implies \forall A B. DM2^{AB} \neg^I$  **by** (*smt MONO-def monI conn*)

**lemma**  $\forall A B. Br(-A) \wedge Br(-B) \longrightarrow DM2^{AB} \neg^I$  **using** *Int-fr-def pB1 conn by auto*

DM2-C

**lemma** *Fr-6*  $\mathcal{F} \implies \forall A B. DM2^{AB} \neg^C$  **by** (*smt MONO-def monC conn*)

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Op(A) \wedge Op(B) \longrightarrow DM2^{AB} \neg^C$  **using** *CF2 ClOpdual EXP-def unfolding conn by auto*

DM3-I

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Cl(A) \wedge Cl(B) \longrightarrow DM3^{AB} \neg^I$  **using** *Int-fr-def pF2 unfolding conn by fastforce*

DM3-C

**lemma** *Fr-1a*  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies \forall A B. DM3^{AB} \neg^C$  **using** *Cl-fr-def Fr-1a-def pF2 unfolding conn by metis*

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Br(A) \wedge Br(B) \longrightarrow DM3^{AB} \neg^C$  **using** *Cl-fr-def pF3 unfolding conn by fastforce*

DM4-I

**lemma** *Fr-1a*  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies \forall A B. DM4^{AB} \neg^I$  **using** *ADDI-a-def Br-fr-def CF1a Int-fr-def pC1 unfolding conn by (metis (full-types))*

**lemma**  $\forall A B. Br(-A) \wedge Br(-B) \longrightarrow DM4^{AB} \neg^I$  **using** *Int-fr-def pB1 conn by auto*

DM4-C

**lemma** *Fr-2*  $\mathcal{F} \implies \forall A B. Op(A) \wedge Op(B) \longrightarrow DM4^{AB} \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by (metis (full-types))*

**lemma** *Fr-1*  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall A B. Fr(A) \wedge Fr(B) \longrightarrow DM4^{AB} \neg^C$  **using** *Cl-fr-def Fr-join-closed Fr-2-def compl-def join-def neg-C-def by auto*

## 8.5 Local contraposition axioms

lCoPw-I

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Cl(A) \longrightarrow lCoPw^{AB}(\rightarrow) \neg^I$  **using** *Int-fr-def OpCldual unfolding conn by auto*

**lemma**  $\forall A B. Br(-B) \longrightarrow lCoPw^{AB}(\rightarrow) \neg^I$  **by** (*simp add: pI1 conn*)

lCoPw-C

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Op(B) \longrightarrow lCoPw^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Br(A) \longrightarrow lCoPw^{AB}(\rightarrow) \neg^C$  **by** (*simp add: pC1 conn*)

lCoP1-I

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Cl(A) \longrightarrow lCoP1^{AB}(\rightarrow) \neg^I$  **using** *Int-fr-def OpCldual unfolding conn by auto*

lCoP1-C

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Op(B) \longrightarrow lCoP1^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Br(A) \longrightarrow lCoP1^{AB}(\rightarrow) \neg^C$  **by** (*simp add: pC1 conn*)

lCoP2-I

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Cl(A) \longrightarrow lCoP2^{AB}(\rightarrow) \neg^I$  **using** *Int-fr-def OpCldual unfolding conn by auto*

**lemma**  $\forall A B. Br(-B) \longrightarrow lCoP2^{AB}(\rightarrow) \neg^I$  **by** (*simp add: pI1 conn*)

lCoP2-C

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Op(B) \longrightarrow lCoP2^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

lCoP3-I

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Cl(A) \longrightarrow lCoP3^{AB}(\rightarrow) \neg^I$  **using** *Int-fr-def OpCldual unfolding conn by auto*

lCoP3-C

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Op(B) \longrightarrow lCoP3^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

## 8.6 Disjunctive syllogism

DS1-I

**lemma**  $\forall A B. DS1^{AB}(\rightarrow) \neg^I$  **by** (*simp add: Int-fr-def conn*)

DS1-C

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Op(A) \longrightarrow DS1^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def Int-fr-def pF2 unfolding conn by fastforce*

DS2-I

**lemma**  $Fr-2 \mathcal{F} \implies \forall A B. Cl(A) \longrightarrow DS2^{AB}(\rightarrow) \neg^I$  **using** *OpCldual unfolding conn by auto*

DS2-C

**lemma**  $\forall A B. DS2^{AB}(\rightarrow) \neg^C$  **using** *Cl-fr-def unfolding conn by auto*

```

end
theory ex-LFIs
  imports topo-negation-conditions
begin
nitpick-params[assms=true, user-axioms=true, show-all, expect=genuine, format=3]

```

## 9 Example application: Logics of Formal Inconsistency (LFIs)

The LFIs [4] [3] are a family of paraconsistent logics featuring a 'consistency' operator  $\circ$  that can be used to recover some classical properties of negation (in particular ECQ). We show how to semantically embed LFIs as extensions of Boolean algebras (here as frontier algebras).

Logical validity can be defined as truth in all worlds/points (i.e. as denoting the top element)

**abbreviation**  $gtrue::\sigma\Rightarrow bool$  ( $[\vdash -]$ ) **where**  $[\vdash A] \equiv \forall w. A w$   
**lemma**  $gtrue-def: [\vdash A] \equiv A \approx \top$  **by** (*simp add: top-def*)

When defining a logic over an existing algebra we have two choices: a global (truth preserving) and a local (truth-degree preserving) notion of logical consequence. For LFIs we prefer the latter.

**abbreviation**  $conseq-global1::\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[- \vdash_g -]$ ) **where**  $[A \vdash_g B] \equiv [\vdash A] \longrightarrow [\vdash B]$   
**abbreviation**  $conseq-global2::\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[-, - \vdash_g -]$ ) **where**  $[A_1, A_2 \vdash_g B] \equiv [\vdash A_1] \wedge [\vdash A_2] \longrightarrow [\vdash B]$   
**abbreviation**  $conseq-global3::\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[-, -, - \vdash_g -]$ ) **where**  $[A_1, A_2, A_3 \vdash_g B] \equiv [\vdash A_1] \wedge [\vdash A_2] \wedge [\vdash A_3] \longrightarrow [\vdash B]$   
**abbreviation**  $conseq-local1::\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[- \vdash -]$ ) **where**  $[A \vdash B] \equiv A \preceq B$   
**abbreviation**  $conseq-local2::\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[-, - \vdash -]$ ) **where**  $[A_1, A_2 \vdash B] \equiv A_1 \wedge A_2 \preceq B$   
**abbreviation**  $conseq-local3::\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow\sigma\Rightarrow bool$  ( $[-, -, - \vdash -]$ ) **where**  $[A_1, A_2, A_3 \vdash B] \equiv A_1 \wedge A_2 \wedge A_3 \preceq B$

For LFIs we use the (paraconsistent) closure-based negation previously defined (taking frontier as primitive).

**abbreviation**  $cneg::\sigma\Rightarrow\sigma$  ( $\neg$ ) **where**  $\neg A \equiv \neg^C A$

In terms of the frontier operator the negation looks as follows:

**lemma**  $\neg A \approx -A \vee \mathcal{F}(-A)$  **by** (*simp add: neg-C-def pC2*)  
**lemma**  $cneg-prop: Fr-2 \mathcal{F} \Longrightarrow \neg A \approx -A \vee \mathcal{F}(A)$  **using** *pC2 pF2 unfolding conn by blast*

This negation is of course boldly paraconsistent.

**lemma**  $[a, \neg a \vdash b]$  **nitpick oops**  
**lemma**  $[a, \neg a \vdash \neg b]$  **nitpick oops**  
**lemma**  $[a, \neg a \vdash_g b]$  **nitpick oops**  
**lemma**  $[a, \neg a \vdash_g \neg b]$  **nitpick oops**

We define two pairs of in/consistency operators and show how they relate to each other. Using LFIs terminology, the minimal logic so encoded corresponds to 'RmbC-ciw' (cf. [3]).

**abbreviation**  $op-inc-a::\sigma\Rightarrow\sigma$  ( $\cdot^A$ - [57]58) **where**  $\cdot^A A \equiv A \wedge \neg A$   
**abbreviation**  $op-con-a::\sigma\Rightarrow\sigma$  ( $\circ^A$ - [57]58) **where**  $\circ^A A \equiv -\cdot^A A$   
**abbreviation**  $op-inc-b::\sigma\Rightarrow\sigma$  ( $\cdot^B$ - [57]58) **where**  $\cdot^B A \equiv \mathcal{B} A$   
**abbreviation**  $op-con-b::\sigma\Rightarrow\sigma$  ( $\circ^B$ - [57]58) **where**  $\circ^B A \equiv \mathcal{B}^c A$

Observe that assuming condition Fr-2 are we allowed to exchange A and B variants.

**lemma** *pincAB*:  $Fr\text{-}2 \mathcal{F} \implies \cdot^A A \approx \cdot^B A$  **using** *Br-fr-def Cl-fr-def pF2 conn* **by** *auto*  
**lemma** *pconAB*:  $Fr\text{-}2 \mathcal{F} \implies \circ^A A \approx \circ^B A$  **using** *pincAB unfolding conn* **by** *simp*

Variants A and B give us slightly different properties.

**lemma** *Prop1*:  $\circ^B A \approx \mathcal{I}^{fp} A$  **using** *fp1 unfolding conn equal-op-def* **by** *metis*  
**lemma**  $\circ^A A \approx A \rightarrow \mathcal{I} A$  **nitpick oops**  
**lemma** *Prop2*:  $Cl A \longleftrightarrow \circ^A -A \approx \top$  **using** *pC2 unfolding conn* **by** *auto*  
**lemma**  $Cl A \rightarrow \circ^B -A \approx \top$  **nitpick oops**  
**lemma** *Prop3*:  $Cl A \longleftrightarrow \cdot^A -A \approx \perp$  **using** *Cl-fr-def unfolding conn* **by** *auto*  
**lemma**  $Cl A \rightarrow \cdot^B -A \approx \perp$  **nitpick oops**  
**lemma** *Prop4*:  $Op A \longleftrightarrow \circ^B A \approx \top$  **using** *Op-Bzero unfolding conn* **by** *simp*  
**lemma**  $Op A \rightarrow \circ^A A \approx \top$  **nitpick oops**  
**lemma** *Prop5*:  $Op A \longleftrightarrow \cdot^B A \approx \perp$  **using** *Op-Bzero* **by** *simp*  
**lemma**  $Op A \rightarrow \cdot^A A \approx \perp$  **nitpick oops**

Importantly, LFIs must satisfy the so-called 'principle of gentle explosion'. Only variant A works here:

**lemma**  $[\circ^A a, a, \neg a \vdash b]$  **using** *compl-def meet-def* **by** *auto*  
**lemma**  $[\circ^A a, a, \neg a \vdash_g b]$  **using** *compl-def meet-def* **by** *auto*  
**lemma**  $[\circ^B a, a, \neg a \vdash b]$  **nitpick oops**  
**lemma**  $[\circ^B a, a, \neg a \vdash_g b]$  **nitpick oops**

In what follows we employ the (minimal) A-variant and verify some properties.

**abbreviation** *op-inc* ::  $\sigma \Rightarrow \sigma (\cdot - [57]58)$  **where**  $\cdot A \equiv \cdot^A A$   
**abbreviation** *op-con* ::  $\sigma \Rightarrow \sigma (\circ - [57]58)$  **where**  $\circ A \equiv \neg \cdot A$

**lemma** *TND*( $\neg$ ) **by** (*simp add: TND-C*)  
**lemma** *ECQm*( $\neg$ ) **nitpick oops**  
**lemma** *Fr-1b*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies LNC(\neg)$  **by** (*simp add: LNC-C PF6*)  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DNI(\neg)$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DNE(\neg)$  **nitpick oops**  
**lemma** *Fr-1b*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies CoPw(\neg)$  **by** (*simp add: CoPw-C PF6*)  
**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP1(\neg)$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP2(\neg)$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP3(\neg)$  **nitpick oops**  
**lemma** *Fr-1a*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies DM3(\neg)$  **by** (*simp add: DM3-C*)  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DM4(\neg)$  **nitpick oops**  
**lemma** *nNor*( $\neg$ ) **by** (*simp add: nNor-C*)  
**lemma** *Fr-3*  $\mathcal{F} \implies nDNor(\neg)$  **by** (*simp add: nDNor-C*)  
**lemma** *Fr-1b*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies MT0(\neg)$  **by** (*simp add: MT0-C PF6*)  
**lemma** *Fr-1b*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies MT1(\neg)$  **by** (*simp add: MT1-C PF6*)  
**lemma**  $\mathfrak{F} \mathcal{F} \implies MT2(\neg)$  **nitpick oops**  
**lemma** *Fr-1b*  $\mathcal{F} \implies Fr\text{-}2 \mathcal{F} \implies Fr\text{-}3 \mathcal{F} \implies MT3(\neg)$  **using** *MT3-C* **by** *auto*

We show how all local contraposition variants (lCoP) can be recovered using the consistency operator. Observe that we can recover in the same way other (weaker) properties: CoP, MT and DNI/DNE (local & global).

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoPw(\rightarrow)(\neg)$  **nitpick oops**  
**lemma** *cons-leop1*:  $[ob, a \rightarrow b \vdash \neg b \rightarrow \neg a]$  **using** *Cl-fr-def conn* **by** *auto*  
**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP1(\rightarrow)(\neg)$  **nitpick oops**  
**lemma** *cons-leop2*:  $[ob, a \rightarrow \neg b \vdash b \rightarrow \neg a]$  **using** *Cl-fr-def conn* **by** *auto*  
**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP2(\rightarrow)(\neg)$  **nitpick oops**  
**lemma** *cons-leop3*:  $[ob, \neg a \rightarrow b \vdash \neg b \rightarrow a]$  **using** *Cl-fr-def conn* **by** *auto*  
**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP3(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *cons-lcop4*:  $[\circ b, \neg a \rightarrow \neg b \vdash b \rightarrow a]$  **using** *Cl-fr-def conn* **by** *auto*

Disjunctive syllogism (DS).

**lemma**  $\mathfrak{F} \mathcal{F} \implies DS1(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *cons-ds1*:  $[\circ a, a \vee b \vdash \neg a \rightarrow b]$  **using** *conn* **by** *auto*

**lemma** *DS2*( $\rightarrow$ )( $\neg$ ) **by** (*metis Cl-fr-def DS2-def compl-def impl-def join-def neg-C-def*)

Further properties.

**lemma**  $[a \wedge \neg a \vdash \neg(\circ a)]$  **by** (*simp add: pC2 conn*)

**lemma**  $\mathfrak{F} \mathcal{F} \implies [\neg(\circ a) \vdash a \wedge \neg a]$  **nitpick oops**

**lemma**  $[\circ a \vdash \neg(a \wedge \neg a)]$  **by** (*simp add: pC2 conn*)

**lemma**  $\mathfrak{F} \mathcal{F} \implies [\neg(a \wedge \neg a) \vdash \circ a]$  **nitpick oops**

The following axioms are commonly employed in the literature on LFIs to obtain stronger logics. We explore under which conditions they can be assumed while keeping the logic boldly paraconsistent.

**abbreviation** *cf* **where**  $cf \equiv DNE(\neg)$

**abbreviation** *ce* **where**  $ce \equiv DNI(\neg)$

**abbreviation** *ciw* **where**  $ciw \equiv \forall P. [\vdash \circ P \vee \cdot P]$

**abbreviation** *ci* **where**  $ci \equiv \forall P. [\neg(\circ P) \vdash \cdot P]$

**abbreviation** *cl* **where**  $cl \equiv \forall P. [\neg(\cdot P) \vdash \circ P]$

**abbreviation** *ca-conj* **where**  $ca-conj \equiv \forall P Q. [\circ P, \circ Q \vdash \circ(P \wedge Q)]$

**abbreviation** *ca-disj* **where**  $ca-disj \equiv \forall P Q. [\circ P, \circ Q \vdash \circ(P \vee Q)]$

**abbreviation** *ca-impl* **where**  $ca-impl \equiv \forall P Q. [\circ P, \circ Q \vdash \circ(P \rightarrow Q)]$

**abbreviation** *ca* **where**  $ca \equiv ca-conj \wedge ca-disj \wedge ca-impl$

*cf*

**lemma**  $\mathfrak{F} \mathcal{F} \implies cf$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \wedge cf \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

*ce*

**lemma**  $\mathfrak{F} \mathcal{F} \implies ce$  **nitpick oops**

**lemma** *Fr-1*  $\mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge ce \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

**lemma** *Fr-1*  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge ce \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

**lemma** *Fr-1a*  $\mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge ce \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

**lemma** *Fr-1b*  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies ce \longrightarrow ECQm(\neg)$  **unfolding Defs using CoP1-XCoP CoP1-def2 CoPw-C DNI-def ECQw-def PF6 XCoP-def2 by auto**

*ciw*

**lemma** *ciw* **by** (*simp add:conn*)

*ci*

**lemma**  $\mathfrak{F} \mathcal{F} \implies ci$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \wedge ci \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

*cl*

**lemma**  $\mathfrak{F} \mathcal{F} \implies cl$  **nitpick oops**

**lemma** *Fr-1*  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cl \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

**lemma** *Fr-1a*  $\mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cl \wedge \sim ECQm(\neg)$  **nitpick[satisfy] oops**

**lemma** *Fr-1b*  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies cl \longrightarrow ECQ(\neg)$  **unfolding Defs by (metis BC-rel Br-Header Br-cl-def bottom-def compl-def eq-ext' meet-def neg-C-def)**

*ca-conj/disj*

```

lemma Fr-1a  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies ca\text{-conj}$  using DM3-C DM3-def conn by auto
lemma Fr-1b  $\mathcal{F} \implies Fr-2 \mathcal{F} \implies ca\text{-disj}$  using ADDI-b-def MONO-ADDIb monI pB1 pincAB unfolding
  conn by metis
lemma  $\mathfrak{F} \mathcal{F} \implies ca\text{-impl}$  nitpick oops

ca-impl

lemma  $ca\text{-impl} \wedge \sim ECQ(\neg)$  oops
lemma  $ca\text{-impl} \longrightarrow ECQm(\neg)$  oops

cf & ci

lemma Fr-1  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cf \wedge ci \wedge \sim ECQm(\neg)$  nitpick[satisfy] oops
lemma Fr-2  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cf \wedge ci \wedge \sim ECQm(\neg)$  nitpick[satisfy] oops
lemma Fr-1b  $\mathcal{F} \wedge Fr-2 \mathcal{F} \wedge cf \wedge ci \wedge \sim ECQ(\neg)$  oops
lemma  $\mathfrak{F} \mathcal{F} \wedge cf \wedge ci \longrightarrow ECQm(\neg)$  oops

cf & cl

lemma Fr-1  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cf \wedge cl \wedge \sim ECQm(\neg)$  nitpick[satisfy] oops
lemma Fr-2  $\mathcal{F} \wedge Fr-3 \mathcal{F} \wedge Fr-4 \mathcal{F} \wedge cf \wedge cl \wedge \sim ECQm(\neg)$  nitpick[satisfy] oops
lemma Fr-1b  $\mathcal{F} \wedge Fr-2 \mathcal{F} \wedge cf \wedge cl \longrightarrow ECQ(\neg)$  unfolding Defs by (smt Br-fr-def Fr-1b-def Prop2
  Prop3 pF3 cneg-prop conn)

end
theory topo-strict-implication
  imports topo-frontier-algebra
begin
nitpick-params[assms=true, user-axioms=true, show-all, expect=genuine, format=3]

```

## 10 Strict implication

We can also employ the closure and interior operations to define different sorts of implications. In this section we preliminary explore a sort of strict implication and check some relevant properties.

A 'strict' implication should entail the classical one. Hence we define it using the interior operator.

```

definition  $imp\text{-I}::\sigma \Rightarrow \sigma \Rightarrow \sigma$  (infix  $\Rightarrow$  51) where  $\alpha \Rightarrow \beta \equiv \mathcal{I}(\alpha \rightarrow \beta)$ 
abbreviation  $imp\text{-I}'::\sigma \Rightarrow \sigma \Rightarrow \sigma$  (infix  $\Leftarrow$  51) where  $\beta \Leftarrow \alpha \equiv \alpha \Rightarrow \beta$ 
declare  $imp\text{-I}\text{-def}$ [conn]

```

```

lemma  $imp\text{-rel}::\forall a b. a \Rightarrow b \preceq a \rightarrow b$  by (simp add: Int-fr-def conn)

```

Under appropriate conditions this implication satisfies a weak variant of the deduction theorem (DT),

```

lemma DTw1:  $\forall a b. a \Rightarrow b \approx \top \longrightarrow a \preceq b$  by (simp add: Int-fr-def conn)
lemma DTw2:  $Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b. a \preceq b \longrightarrow a \Rightarrow b \approx \top$  using IF3 dNOR-def unfolding
  conn by auto

```

and a variant of modus ponens and modus tollens resp.

```

lemma MP:  $\forall a b. a \wedge (a \Rightarrow b) \preceq b$  by (simp add: Int-fr-def conn)
lemma MT:  $\forall a b. (a \Rightarrow b) \wedge \neg b \preceq \neg a$  using MP conn by auto

```

However the full DT (actually right-to-left: implication introduction) is not valid.



**lemma DT1:**  $\forall a b X. X \preceq a \Rightarrow b \longrightarrow X \wedge a \preceq b$  **by** (*simp add: Int-fr-def conn*)

**lemma DT2:**  $\wp \mathcal{F} \Longrightarrow \forall a b X. X \wedge a \preceq b \longrightarrow X \preceq a \Rightarrow b$  **nitpick oops**

This implication has thus a sort of 'relevant' behaviour. For instance, the following are not valid:

**lemma**  $\wp \mathcal{F} \Longrightarrow \forall a b. (a \Rightarrow (b \Rightarrow a)) \approx \top$  **nitpick oops**

**lemma**  $\wp \mathcal{F} \Longrightarrow \forall a b. (a \Rightarrow ((a \Rightarrow b) \Rightarrow b)) \approx \top$  **nitpick oops**

**lemma**  $\wp \mathcal{F} \Longrightarrow \forall a b c. (a \Rightarrow b) \vee (b \Rightarrow c) \approx \top$  **nitpick oops**

**lemma**  $\wp \mathcal{F} \Longrightarrow \forall a b. ((a \Rightarrow b) \Rightarrow a) \Rightarrow a \approx \top$  **nitpick oops**

In contrast the properties below are valid for appropriate conditions.

**lemma iprop0:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a. a \Rightarrow a \approx \top$  **using DTw2 pI2 by fastforce**

**lemma iprop1:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b. a \wedge (a \Rightarrow b) \Rightarrow b \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop2:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b. a \Rightarrow (b \Rightarrow b) \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop3:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b. ((a \Rightarrow a) \Rightarrow b) \Rightarrow b \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop4:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b. (a \wedge b) \Rightarrow a \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop5:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b. a \Rightarrow (a \vee b) \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop6a:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b c. (a \wedge (b \vee c)) \Rightarrow ((a \wedge b) \vee (a \wedge c)) \approx \top$  **using DTw2 pI2 unfolding conn by fastforce**

**lemma iprop6b:**  $Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b c. (a \wedge (b \vee c)) \Leftarrow ((a \wedge b) \vee (a \wedge c)) \approx \top$  **using DTw2 unfolding conn by fastforce**

**lemma iprop7':**  $Fr-1 \mathcal{F} \Longrightarrow \forall a b c. (a \Rightarrow b) \wedge (b \Rightarrow c) \preceq (a \Rightarrow c)$  **proof –**  
**assume fr1:**  $Fr-1 \mathcal{F}$

{ **fix**  $a b c$

**have**  $(a \rightarrow b) \wedge (b \rightarrow c) \preceq (a \rightarrow c)$  **unfolding conn by simp**

**hence**  $\mathcal{I}((a \rightarrow b) \wedge (b \rightarrow c)) \preceq \mathcal{I}(a \rightarrow c)$  **using MONO-def PF1 fr1 monI by simp**

**moreover from fr1 have**  $let A=(a \rightarrow b); B=(b \rightarrow c)$  **in**  $\mathcal{I}(A \wedge B) \approx \mathcal{I} A \wedge \mathcal{I} B$  **using IF1**

**MULT-def by simp**

**ultimately have**  $\mathcal{I}(a \rightarrow b) \wedge \mathcal{I}(b \rightarrow c) \preceq \mathcal{I}(a \rightarrow c)$  **unfolding conn by simp**

**} thus ?thesis unfolding conn by blast**

**qed**

**lemma iprop7:**  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b c. ((a \Rightarrow b) \wedge (b \Rightarrow c)) \Rightarrow (a \Rightarrow c) \approx \top$   
**by** (*simp add: DTw2 iprop7'*)

**lemma iprop8a':**  $Fr-1 \mathcal{F} \Longrightarrow \forall a b c. (a \Rightarrow b) \wedge (a \Rightarrow c) \preceq a \Rightarrow (b \wedge c)$  **proof –**

**assume fr1:**  $Fr-1 \mathcal{F}$

{ **fix**  $a b c$

**have**  $(a \rightarrow b) \wedge (a \rightarrow c) \preceq (a \rightarrow (b \wedge c))$  **unfolding conn by simp**

**hence**  $\mathcal{I}((a \rightarrow b) \wedge (a \rightarrow c)) \preceq \mathcal{I}(a \rightarrow (b \wedge c))$  **using MONO-def PF1 fr1 monI by simp**

**moreover from fr1 have**  $let A=(a \rightarrow b); B=(a \rightarrow c)$  **in**  $\mathcal{I}(A \wedge B) \approx \mathcal{I} A \wedge \mathcal{I} B$  **using IF1**

**MULT-def by simp**

**ultimately have**  $\mathcal{I}(a \rightarrow b) \wedge \mathcal{I}(a \rightarrow c) \preceq \mathcal{I}(a \rightarrow (b \wedge c))$  **unfolding conn by simp**

**} thus ?thesis unfolding conn by simp**

**qed**

**lemma iprop8b':**  $Fr-1b \mathcal{F} \Longrightarrow \forall a b c. (a \Rightarrow b) \wedge (a \Rightarrow c) \succeq a \Rightarrow (b \wedge c)$  **by** (*smt MONO-def monI conn*)

**lemma iprop8a:**  $Fr-1 \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b c. ((a \Rightarrow b) \wedge (a \Rightarrow c)) \Rightarrow (a \Rightarrow (b \wedge c)) \approx \top$   
**by** (*simp add: DTw2 iprop8a'*)

**lemma iprop8b:**  $Fr-1b \mathcal{F} \Longrightarrow Fr-2 \mathcal{F} \Longrightarrow Fr-3 \mathcal{F} \Longrightarrow \forall a b c. ((a \Rightarrow b) \wedge (a \Rightarrow c)) \Leftarrow (a \Rightarrow (b \wedge c))$

$\approx \top$  by (simp add: DTw2 iprop8b')

**lemma iprop9a':**  $Fr-1 \mathcal{F} \implies \forall a b c. ((a \implies b) \wedge (c \implies b)) \preceq ((a \vee c) \implies b)$  **proof** –

assume  $fr1: Fr-1 \mathcal{F}$

{ fix  $a b c$

have  $(a \rightarrow b) \wedge (c \rightarrow b) \preceq (a \vee c) \rightarrow b$  **unfolding conn by simp**

hence  $\mathcal{I}((a \rightarrow b) \wedge (c \rightarrow b)) \preceq \mathcal{I}((a \vee c) \rightarrow b)$  **using MONO-def PF1 fr1 monI by simp**

moreover from  $fr1$  have let  $A=(a \rightarrow b); B=(c \rightarrow b)$  in  $\mathcal{I}(A \wedge B) \approx \mathcal{I} A \wedge \mathcal{I} B$  **using IF1**

**MULT-def by simp**

ultimately have  $\mathcal{I}(a \rightarrow b) \wedge \mathcal{I}(c \rightarrow b) \preceq \mathcal{I}((a \vee c) \rightarrow b)$  **unfolding conn by simp**

} thus ?thesis **unfolding conn by simp**

qed

**lemma iprop9b':**  $Fr-1b \mathcal{F} \implies \forall a b c. ((a \implies b) \wedge (c \implies b)) \succeq ((a \vee c) \implies b)$  **by (smt MONO-def monI conn)**

**lemma iprop9a:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b c. ((a \implies b) \wedge (c \implies b)) \implies ((a \vee c) \implies b) \approx \top$  **by (simp add: DTw2 iprop9a')**

**lemma iprop9b:**  $Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b c. ((a \implies b) \wedge (c \implies b)) \Leftarrow ((a \vee c) \implies b) \approx \top$  **by (simp add: DTw2 iprop9b')**

**lemma iprop10':**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall a b c. a \implies (b \implies c) \preceq (a \implies b) \implies (a \implies c)$  **proof** –

assume  $fr1: Fr-1 \mathcal{F}$  and  $fr2: Fr-2 \mathcal{F}$  and  $fr4: Fr-4 \mathcal{F}$

{ fix  $a b c$

have  $a \rightarrow (b \rightarrow c) \preceq (a \rightarrow b) \rightarrow (a \rightarrow c)$  **unfolding conn by simp**

hence  $a \rightarrow (b \implies c) \preceq (a \rightarrow b) \rightarrow (a \rightarrow c)$  **using Int-fr-def conn by auto**

hence  $\mathcal{I}(a \rightarrow (b \implies c)) \preceq \mathcal{I}((a \rightarrow b) \rightarrow (a \rightarrow c))$  **using MONO-def PF1 fr1 monI by simp**

moreover from  $fr1$  have let  $A=(a \rightarrow b); B=(a \rightarrow c)$  in  $\mathcal{I}(A \rightarrow B) \preceq \mathcal{I} A \rightarrow \mathcal{I} B$  **using IF1**

**Int-7-def PI7 by auto**

ultimately have  $\mathcal{I}(a \rightarrow (b \implies c)) \preceq \mathcal{I}(a \rightarrow b) \rightarrow \mathcal{I}(a \rightarrow c)$  **by (metis (full-types))**

hence  $\mathcal{I}(\mathcal{I}(a \rightarrow (b \implies c))) \preceq \mathcal{I}(\mathcal{I}(a \rightarrow b) \rightarrow \mathcal{I}(a \rightarrow c))$  **using MONO-def PF1 fr1 monI by simp**

hence  $\mathcal{I}(a \rightarrow (b \implies c)) \preceq \mathcal{I}(\mathcal{I}(a \rightarrow b) \rightarrow \mathcal{I}(a \rightarrow c))$  **using Int-Open PF1 fr1 fr2 fr4 by blast**

hence  $(a \implies (b \implies c)) \preceq (a \implies b) \implies (a \implies c)$  **using Int-Open PF1 fr1 fr2 fr4 unfolding conn by**

**blast**

} thus ?thesis **by blast**

qed

**lemma iprop10:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall a b c. (a \implies (b \implies c)) \implies ((a \implies b) \implies (a \implies c)) \approx \top$  **by (simp add: DTw2 iprop10')**

**lemma  $\wp \mathcal{F} \implies \forall a b c. a \implies (b \implies c) \succeq (a \implies b) \implies (a \implies c)$  nitpick oops**

**lemma iprop11a':**  $Fr-1 \mathcal{F} \implies \forall a b. (a \implies (a \implies b)) \preceq (a \implies b)$  **by (smt MONO-def PF1 imp-rel monI conn)**

**lemma iprop11b':**  $\wp \mathcal{F} \implies \forall a b. (a \implies (a \implies b)) \succeq (a \implies b)$  **unfolding PF1 by (metis Int-Open MONO-def imp-I-def impl-def monI)**

**lemma iprop11a:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b. (a \implies (a \implies b)) \implies (a \implies b) \approx \top$  **using DTw2 iprop11a' by blast**

**lemma iprop11b:**  $\wp \mathcal{F} \implies \forall a b. (a \implies (a \implies b)) \Leftarrow (a \implies b) \approx \top$  **using DTw2 iprop11b' by blast**

In particular, 'strengthening the antecedent' is valid only under certain conditions:

**lemma SA':**  $Fr-1b \mathcal{F} \implies \forall a b c. a \implies c \preceq (a \wedge b) \implies c$  **by (smt MONO-def monI conn)**

**lemma SA:**  $Fr-1b \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b c. (a \implies c) \implies ((a \wedge b) \implies c) \approx \top$  **using SA' using DTw2 by fastforce**

**lemma Fr-1a:**  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies Fr-4 \mathcal{F} \implies \forall a b c. a \implies c \preceq (a \wedge b) \implies c$  **nitpick oops**

**lemma** *Fr-1a*  $\mathcal{F} \implies \text{Fr-2 } \mathcal{F} \implies \text{Fr-3 } \mathcal{F} \implies \text{Fr-4 } \mathcal{F} \implies \forall a b c. (a \implies c) \implies ((a \wedge b) \implies c) \approx \top$  **nitpick oops**

Similarly, 'weakening the consequent' is valid only under certain conditions.

**lemma** *WC'*: *Fr-1b*  $\mathcal{F} \implies \forall a b c. a \implies c \preceq a \implies (c \vee b)$  **by** (*smt MONO-def monI conn*)

**lemma** *WC*: *Fr-1b*  $\mathcal{F} \implies \text{Fr-2 } \mathcal{F} \implies \text{Fr-3 } \mathcal{F} \implies \forall a b c. (a \implies c) \implies (a \implies (c \vee b)) \approx \top$  **using** *DTw2 WC'* **by** *fastforce*

**lemma** *Fr-1a*  $\mathcal{F} \implies \text{Fr-2 } \mathcal{F} \implies \text{Fr-3 } \mathcal{F} \implies \text{Fr-4 } \mathcal{F} \implies \forall a b c. a \implies c \preceq a \implies (c \vee b)$  **nitpick oops**

**lemma** *Fr-1a*  $\mathcal{F} \implies \text{Fr-2 } \mathcal{F} \implies \text{Fr-3 } \mathcal{F} \implies \text{Fr-4 } \mathcal{F} \implies \forall a b c. (a \implies c) \implies (a \implies (c \vee b)) \approx \top$  **nitpick oops**

**end**

**theory** *ex-subminimal-logics*

**imports** *topo-negation-conditions topo-strict-implication*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 11 Example application: Subintuitionistic and subminimal logics

In this section we examine some special paracomplete logics. The idea is to illustrate an approach by means of which we can obtain logics which are boldly paracomplete and (non-boldly) paraconsistent at the same time, Johansson's 'minimal logic' [7] being the paradigmatic case we aim at modelling.

Drawing upon the literature on Johanson's minimal logic, we introduce an unconstrained propositional constant  $Q$ , which we then employ to define a 'rigid' frontier operation  $\mathcal{F}'$ .

**consts**  $Q::\sigma$

**abbreviation**  $\mathcal{F}' \equiv \lambda X. Q$

**abbreviation**  $\mathcal{I}' \equiv \mathcal{I}_F \mathcal{F}'$

**abbreviation**  $\mathcal{C}' \equiv \mathcal{C}_F \mathcal{F}'$

**abbreviation**  $\mathcal{B}' \equiv \mathcal{B}_F \mathcal{F}'$

As defined,  $\mathcal{F}'$  (and its corresponding closure operation) satisfies several semantic conditions.

**lemma** *Fr-1*  $\mathcal{F}' \wedge \text{Fr-2 } \mathcal{F}' \wedge \text{Fr-4 } \mathcal{F}'$  **by** (*simp add: Fr-1-def Fr-2-def Fr-4-def conn*)

**lemma** *Cl-1*  $\mathcal{C}' \wedge \text{Cl-2 } \mathcal{C}' \wedge \text{Cl-4 } \mathcal{C}'$  **using** *ADDI-def CF2 IDEMb-def Cl-fr-def PC4* **unfolding** *conn* **by** *auto*

However *Fr-3* is not valid. In fact, adding it by hand would collapse into classical logic (making all sets clopen).

**lemma** *Fr-3*  $\mathcal{F}'$  **nitpick oops**

**lemma** *Cl-3*  $\mathcal{C}'$  **nitpick oops**

**lemma** *Fr-3*  $\mathcal{F}' \implies \forall A. \mathcal{F}'(A) \approx \perp$  **by** (*simp add: NOR-def*)

In order to obtain a paracomplete logic not validating ECQ, we define negation as follows,

**abbreviation** *neg-IC*:  $\sigma \implies \sigma$  ( $\neg$ ) **where**  $\neg A \equiv \mathcal{C}'(\mathcal{I}(-A))$

and observe that some plausible semantic properties obtain:

**lemma** *Q-def1*:  $\forall A. Q \approx \neg A \wedge \neg(\neg A)$  **using** *Cl-fr-def IF2 dEXP-def conn* **by** *auto*

**lemma** *Q-def2*:  $Fr-1b \mathcal{F} \implies \forall A. Q \approx \neg(A \vee \neg A)$  **by** (*smt Cl-fr-def IF2 dEXP-def MONO-def monI conn*)

**lemma** *neg-Idef*:  $\forall A. \neg A \approx \mathcal{I}(\neg A) \vee Q$  **by** (*simp add: Cl-fr-def*)

**lemma** *neg-Cdef*:  $Fr-2 \mathcal{F} \implies \forall A. \neg A \approx \mathcal{C}(A) \rightarrow Q$  **using** *Cl-fr-def Fr-2-def Int-fr-def conn* **by** *auto*

The negation so defined validates some properties corresponding to a (rather weak) paraconsistent logic:

**lemma**  $\mathfrak{F} \mathcal{F} \implies TND \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies TNDw \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies TNDm \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies ECQ \neg$  **nitpick oops**

**lemma** *ECQw*:  $ECQw \neg$  **using** *Cl-fr-def Disj-I ECQw-def unfolding conn* **by** *auto*

**lemma** *ECQm*:  $ECQm \neg$  **using** *Cl-fr-def Disj-I ECQm-def unfolding conn* **by** *auto*

**lemma**  $\mathfrak{F} \mathcal{F} \implies LNC \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNI \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies DNE \neg$  **nitpick oops**

**lemma** *CoPw*:  $Fr-1b \mathcal{F} \implies CoPw \neg$  **using** *Cl-fr-def MONO-def monI unfolding Defs conn* **by** *smt*

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP1 \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP2 \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies CoP3 \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies XCoP \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies DM3 \neg$  **nitpick oops**

**lemma** *DM4*:  $Fr-1a \mathcal{F} \implies DM4 \neg$  **using** *ADDI-a-def Cl-fr-def DM4-def IC1b IF1b dual-def unfolding conn* **by** *smt*

**lemma** *Nor*:  $Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies nNor \neg$  **using** *Cl-fr-def nNor-I nNor-def unfolding conn* **by** *auto*

**lemma**  $\mathfrak{F} \mathcal{F} \implies nDNor \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoPw(\rightarrow) \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP1(\rightarrow) \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP2(\rightarrow) \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP3(\rightarrow) \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies DS1(\rightarrow) \neg$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies DS2(\rightarrow) \neg$  **nitpick oops**

Moreover, we cannot have both DNI and DNE without validating ECQ (thus losing paraconsistency).

**lemma**  $DNI \neg \wedge DNE \neg \longrightarrow ECQ \neg$  **using** *DNE-def ECQ-def Int-fr-def neg-Idef unfolding conn* **by** (*metis (no-types, lifting)*)

However, we can have all of De Morgan laws while keeping (non-bold) paraconsistency.

**lemma**  $\sim ECQ \neg \wedge DM1 \neg \wedge DM2 \neg \wedge DM3 \neg \wedge DM4 \neg \wedge \mathfrak{F} \mathcal{F}$  **nitpick**[*satisfy, card w=3*] **oops**

Below we combine negation with strict implication and verify some interesting properties. For instance, the following are not valid (and cannot become valid by adding semantic restrictions).

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (\neg a \implies (a \implies b)) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (\neg a \rightarrow (a \rightarrow b)) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \wedge \neg a \implies b) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \wedge \neg a \rightarrow b) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \implies (b \vee \neg b)) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \rightarrow (b \vee \neg b)) \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (a \implies \neg a) \implies \neg a \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (a \rightarrow \neg a) \rightarrow \neg a \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \wedge \neg a) \implies b \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \wedge \neg a) \rightarrow b \approx \top$  **nitpick oops**

**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. a \implies (b \vee \neg b) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. a \rightarrow (b \vee \neg b) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \leftrightarrow b) \implies (\neg a \leftrightarrow \neg b) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \leftrightarrow b) \rightarrow (\neg a \leftrightarrow \neg b) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \implies b) \wedge (a \implies \neg b) \implies \neg a \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \rightarrow b) \wedge (a \rightarrow \neg b) \implies \neg a \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (\neg a \implies \perp) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (\neg a \rightarrow \perp) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (\neg a \implies \neg(\neg\top)) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. (\neg a \rightarrow \neg(\neg\top)) \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. \neg(\neg(\neg a)) \implies \neg a \approx \top$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies \forall a. \neg(\neg(\neg a)) \rightarrow \neg a \approx \top$  **nitpick oops**

The (weak) local contraposition axiom is indeed valid under appropriate conditions.

**lemma**  $lCoPw$ :  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies Fr-4 \mathcal{F} \implies lCoPw(\implies) \neg$  **proof** –  
**assume**  $fr1$ :  $Fr-1 \mathcal{F}$  and  $fr2$ :  $Fr-2 \mathcal{F}$  and  $fr3$ :  $Fr-3 \mathcal{F}$  and  $fr4$ :  $Fr-4 \mathcal{F}$   
**{ fix**  $a b$   
**from**  $fr2$  **have**  $\neg b \rightarrow \neg a \approx (C a \rightarrow C b) \vee Q$  **using**  $Cl-fr-def$   $Fr-2-def$   $Int-fr-def$   $conn$  **by**  $auto$   
**moreover from**  $fr1$   $fr2$   $fr3$  **have**  $\mathcal{I}(a \rightarrow b) \preceq C a \rightarrow C b$  **using**  $IC-imp$  **by**  $simp$   
**ultimately have**  $\mathcal{I}(a \rightarrow b) \preceq \neg b \rightarrow \neg a$  **unfolding**  $conn$  **by**  $simp$   
**moreover from**  $fr1$   $fr2$   $fr4$  **have** **let**  $A=(a \rightarrow b)$ ;  $B=(\neg b \rightarrow \neg a)$  **in**  $\mathcal{I} A \preceq B \longrightarrow \mathcal{I} A \preceq \mathcal{I} B$   
**using**  $PF1$   $MONO-MULTa$   $IF1a$   $IF4$   $PI9$   $Int-9-def$  **by**  $smt$   
**ultimately have**  $\mathcal{I}(a \rightarrow b) \preceq \mathcal{I}(\neg b \rightarrow \neg a)$  **by**  $simp$   
**} hence**  $lCoPw(\implies) \neg$  **unfolding**  $Defs$   $conn$  **by**  $blast$   
**thus**  $?thesis$  **by**  $simp$

**qed**

**lemma**  $lCoPw-strict$ :  $\mathfrak{F} \mathcal{F} \implies \forall a b. (a \implies b) \implies (\neg b \implies \neg a) \approx \top$  **by** ( $metis$  ( $no-types$ ,  $lifting$ )  $DTw2$   $lCoPw$   $lCoPw-def$ )

However, other (local) contraposition axioms are not valid.

**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP1(\implies) \neg$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP2(\implies) \neg$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies lCoP3(\implies) \neg$  **nitpick oops**

And this time no variant of disjunctive syllogism is valid.

**lemma**  $\mathfrak{F} \mathcal{F} \implies DS1(\implies) \neg$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DS2(\implies) \neg$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DS2(\implies) \neg$  **nitpick oops**  
**lemma**  $\mathfrak{F} \mathcal{F} \implies DS4(\implies) \neg$  **nitpick oops**

Interestingly, one of the local contraposition axioms ( $lCoP1$ ) follows from DNI.

**lemma**  $DNI-lCoP1$ :  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies Fr-4 \mathcal{F} \implies DNI \neg \longrightarrow lCoP1(\implies) \neg$  **proof** –  
**assume**  $fr1$ :  $Fr-1 \mathcal{F}$  and  $fr2$ :  $Fr-2 \mathcal{F}$  and  $fr3$ :  $Fr-3 \mathcal{F}$  and  $fr4$ :  $Fr-4 \mathcal{F}$   
**{ assume**  $dni$ :  $DNI \neg$   
**{ fix**  $a b$   
**from**  $fr1$   $fr2$   $fr3$   $fr4$  **have**  $lCoPw(\implies) \neg$  **using**  $lCoPw$  **by**  $simp$   
**hence**  $1$ :  $a \implies \neg b \preceq \neg(\neg b) \implies \neg a$  **unfolding**  $lCoPw-def$  **by**  $simp$   
**from**  $fr1$  **have**  $2$ : **let**  $A=b$ ;  $B=\neg(\neg b)$ ;  $C=\neg a$  **in**  $A \preceq B \longrightarrow \mathcal{I}(B \rightarrow C) \preceq \mathcal{I}(A \rightarrow C)$  **by** ( $simp$   $add$ :  $MONO-ant$   $PF1$   $monI$ )  
**from**  $dni$  **have**  $dnib$ :  $b \preceq \neg(\neg b)$  **unfolding**  $DNI-def$  **by**  $simp$   
**from**  $1$   $2$   $dnib$  **have**  $a \implies \neg b \preceq b \implies \neg a$  **unfolding**  $conn$  **by**  $meson$   
**} hence**  $lCoP1(\implies) \neg$  **unfolding**  $Defs$  **by**  $blast$   
**} thus**  $?thesis$  **by**  $simp$   
**qed**

This entails some other interesting results.

**lemma** *DNI-CoP1*:  $Fr-1b \mathcal{F} \implies DNI \neg \implies CoP1 \neg$  **using** *CoP1-def2 CoPw* **by** *blast*

**lemma** *CoP1-LNC*:  $CoP1 \neg \implies LNC \neg$  **using** *CoP1-def ECQm-def LNC-def Cl-fr-def Disj-I ECQm-def unfolding conn* **by** *smt*

**lemma** *DNI-LNC*:  $Fr-1b \mathcal{F} \implies DNI \neg \implies LNC \neg$  **by** (*simp add: CoP1-LNC DNI-CoP1*)

The following variants of modus tollens also obtain.

**lemma** *MT*:  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b. (a \implies b) \wedge \neg b \preceq \neg a$  **using** *Cl-fr-def Fr-2-def IC-imp Int-fr-def* **unfolding** *conn* **by** *metis*

**lemma** *MT'*:  $Fr-1 \mathcal{F} \implies Fr-2 \mathcal{F} \implies Fr-3 \mathcal{F} \implies \forall a b. ((a \implies b) \wedge \neg b) \implies \neg a \approx \top$  **by** (*simp add: DTw2 MT*)

We now semantically characterize (an approximation of) Johansson's Minimal Logic along with some exemplary 'subminimal' logics (observing that many more are possible). We check some relevant properties.

**abbreviation** *JML*  $\equiv \mathfrak{F} \mathcal{F} \wedge DNI \neg$

**abbreviation** *SML1*  $\equiv \mathfrak{F} \mathcal{F}$

**abbreviation** *SML2*  $\equiv Fr-1 \mathcal{F} \wedge Fr-2 \mathcal{F} \wedge Fr-3 \mathcal{F}$

**abbreviation** *SML3*  $\equiv Fr-1 \mathcal{F}$

**abbreviation** *SML4*  $\equiv Fr-1b \mathcal{F}$

TND:

**lemma** *JML*  $\implies TND \neg$  **nitpick oops**

**lemma** *JML*  $\implies TNDw \neg$  **nitpick oops**

**lemma** *JML*  $\implies TNDm \neg$  **nitpick oops**

ECQ:

**lemma** *JML*  $\implies ECQ \neg$  **nitpick oops**

**lemma** *ECQw*  $\neg$  **using** *Cl-fr-def Disj-I ECQw-def* **unfolding** *conn* **by** *auto*

**lemma** *ECQm*  $\neg$  **using** *Cl-fr-def Disj-I ECQm-def* **unfolding** *conn* **by** *auto*

LNC:

**lemma** *JML*  $\implies LNC \neg$  **using** *DNI-LNC PF1* **by** *blast*

**lemma** *SML1*  $\implies LNC \neg$  **nitpick oops**

(r)DNI/DNE:

**lemma** *JML*  $\implies DNI \neg$  **using** *CoP1-def2* **by** *blast*

**lemma** *SML1*  $\implies rDNI \neg$  **nitpick oops**

**lemma** *JML*  $\implies rDNE \neg$  **nitpick oops**

CoP/MT:

**lemma** *SML4*  $\implies CoPw \neg$  **unfolding** *Defs* **by** (*smt Cl-fr-def MONO-def monI conn*)

**lemma** *JML*  $\implies CoP1 \neg$  **using** *DNI-CoP1 PF1* **by** *blast*

**lemma** *SML1*  $\implies MT1 \neg$  **nitpick oops**

**lemma** *JML*  $\implies MT2 \neg$  **nitpick oops**

**lemma** *JML*  $\implies MT3 \neg$  **nitpick oops**

XCoP:

**lemma** *JML*  $\implies XCoP \neg$  **nitpick oops**

DM3/4:

**lemma** *JML*  $\implies DM3 \neg$  **nitpick oops**

**lemma** *SML3*  $\implies DM_4 \neg$  **by** (*simp add: DM\_4 PF1*)

**lemma** *SML4*  $\implies DM_4 \neg$  **nitpick oops**

nNor/nDNor:

**lemma** *SML2*  $\implies nNor \neg$  **using** *Cl-fr-def nNor-I nNor-def* **unfolding** *conn* **by** *auto*

**lemma** *SML3*  $\implies nNor \neg$  **nitpick oops**

**lemma** *JML*  $\implies nDNor \neg$  **nitpick oops**

lCoP classical:

**lemma** *JML*  $\implies lCoPw(\rightarrow) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP1(\rightarrow) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP2(\rightarrow) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP3(\rightarrow) \neg$  **nitpick oops**

DS classical:

**lemma** *JML*  $\implies DS1(\rightarrow) \neg$  **nitpick oops**

**lemma** *JML*  $\implies DS2(\rightarrow) \neg$  **nitpick oops**

lCoP strict:

**lemma** *SML1*  $\implies lCoPw(\implies) \neg$  **using** *lCoPw* **by** *blast*

**lemma** *SML2*  $\implies lCoPw(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP1(\implies) \neg$  **using** *CoP1-def2 DNI-lCoP1* **by** *blast*

**lemma** *SML1*  $\implies lCoP1(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP2(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lCoP3(\implies) \neg$  **nitpick oops**

lMT strict:

**lemma** *SML2*  $\implies lMT0(\implies) \neg$  **unfolding** *Defs* **using** *MT* **by** *auto*

**lemma** *SML3*  $\implies lMT0(\implies) \neg$  **oops**

**lemma** *SML4*  $\implies lMT0(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lMT1(\implies) \neg$  **by** (*smt DNI-lCoP1 DT1 lCoP1-def lMT1-def*)

**lemma** *SML1*  $\implies lMT1(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lMT2(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies lMT3(\implies) \neg$  **nitpick oops**

DS strict:

**lemma** *JML*  $\implies DS1(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies DS2(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies DS3(\implies) \neg$  **nitpick oops**

**lemma** *JML*  $\implies DS4(\implies) \neg$  **nitpick oops**

**end**

**theory** *topo-derivative-algebra*

**imports** *topo-operators-derivative*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 12 Derivative algebra

The closure of a set  $A$  ( $\mathcal{C}(A)$ ) can be seen as the set  $A$  augmented by (i) its boundary points, or (ii) its accumulation/limit points. We explore the second variant by drawing on the notion of derivative algebra.

Declares a primitive (unconstrained) derivative (aka. derived-set) operation and defines others from it.

**consts**  $\mathcal{D}::\sigma\Rightarrow\sigma$   
**abbreviation**  $\mathcal{I} \equiv \mathcal{I}_D \mathcal{D}$  — interior  
**abbreviation**  $\mathcal{C} \equiv \mathcal{C}_D \mathcal{D}$  — closure  
**abbreviation**  $\mathcal{B} \equiv \mathcal{B}_D \mathcal{D}$  — border  
**abbreviation**  $\mathcal{F} \equiv \mathcal{F}_D \mathcal{D}$  — frontier  
**abbreviation**  $\mathcal{K} \equiv \mathcal{K}_D \mathcal{D}$  — coherence

## 12.1 Basic properties

Verifies minimal conditions under which operators resulting from conversion functions coincide.

**lemma**  $ICdual: \mathcal{I} \equiv \mathcal{C}^d$  **by** (*simp add: dual-der2 equal-op-def*)  
**lemma**  $ICdual': \mathcal{C} \equiv \mathcal{I}^d$  **by** (*simp add: dual-der1 equal-op-def*)  
**lemma**  $BI-rel: \mathcal{B} \equiv \mathcal{B}_I \mathcal{I}$  **using**  $Br-der-def Br-int-def Int-der-def$  **unfolding**  $equal-op-def conn$  **by** *auto*  
**lemma**  $IB-rel: \mathcal{I} \equiv \mathcal{I}_B \mathcal{B}$  **using**  $Br-der-def Int-br-def Int-der-def$  **unfolding**  $equal-op-def conn$  **by** *auto*  
**lemma**  $BC-rel: \mathcal{B} \equiv \mathcal{B}_C \mathcal{C}$  **using**  $BI-BC-rel BI-rel dual-der1$  **by** *auto*  
**lemma**  $CB-rel: \mathcal{C} \equiv \mathcal{C}_B \mathcal{B}$  **using**  $Br-der-def2 Cl-br-def Int-der-def2 dual-def dual-der1$  **unfolding**  $equal-op-def conn$  **by** *auto*  
**lemma**  $FI-rel: \mathcal{F} \equiv \mathcal{F}_I \mathcal{I}$  **by** (*metis Cl-der-def FI2 Fr-2-def Fr-der-def2 Fr-int-def ICdual' dual-def eq-ext' equal-op-def*)  
**lemma**  $FC-rel: \mathcal{F} \equiv \mathcal{F}_C \mathcal{C}$  **by** (*simp add: Cl-der-def Fr-cl-def Fr-der-def2 equal-op-def*)  
**lemma**  $FB-rel: \mathcal{F} \equiv \mathcal{F}_B \mathcal{B}$  **by** (*smt Br-der-def CB-rel Cl-br-def Cl-der-def Fr-br-def Fr-der-def Fr-der-def2 equal-op-def conn*)

Recall that derivative and coherence operations cannot be obtained from either interior, closure, border nor frontier. The derivative operation can indeed be seen as being more fundamental than the other ones.

Fixed-point and other operators are interestingly related.

**lemma**  $fp1: \mathcal{I}^{fp} \equiv \mathcal{B}^c$  **by** (*smt BI-rel Br-int-def IB-rel Int-br-def equal-op-def conn*)  
**lemma**  $fp2: \mathcal{B}^{fp} \equiv \mathcal{I}^c$  **using**  $Br-der-def Int-der-def$  **unfolding**  $equal-op-def conn$  **by** *auto*  
**lemma**  $fp3: \mathcal{C}^{fp} \equiv \mathcal{B}^d$  **by** (*smt BI-rel Br-int-def CB-rel Cl-br-def dual-def equal-op-def conn*)  
**lemma**  $fp4: (\mathcal{B}^d)^{fp} \equiv \mathcal{C}$  **by** (*smt dimp-def equal-op-def fp3*)  
**lemma**  $fp5: \mathcal{F}^{fp} \equiv \mathcal{B} \sqcup (\mathcal{C}^c)$  **using**  $Br-der-def Cl-der-def Fr-der-def$  **unfolding**  $equal-op-def conn$  **by** *auto*  
**lemma**  $fp6: \mathcal{D}^{fp} \equiv \mathcal{K} \sqcup (\mathcal{C}^c)$  **using**  $Cl-der-def Kh-der-def equal-op-def conn$  **by** *fastforce*

Different inter-relations (some redundant ones are kept to help the provers).

**lemma**  $monI: Der-1b \mathcal{D} \Longrightarrow MONO \mathcal{I}$  **by** (*simp add: ID1b MONO-MULTa*)  
**lemma**  $monC: Der-1b \mathcal{D} \Longrightarrow MONO \mathcal{C}$  **by** (*simp add: CD1b MONO-ADDIb*)  
**lemma**  $pB1: \forall A. \mathcal{B} A \approx A \leftarrow \mathcal{I} A$  **using**  $BI-rel Br-int-def eq-ext'$  **by** *fastforce*  
**lemma**  $pB2: \forall A. \mathcal{B} A \approx A \wedge \mathcal{F} A$  **using**  $Br-der-def Fr-der-def conn$  **by** *auto*  
**lemma**  $pB3: \forall A. \mathcal{B}(-A) \approx -A \wedge \mathcal{F} A$  **using**  $FD2 Fr-2-def meet-def pB2$  **by** *auto*  
**lemma**  $pB4: \forall A. \mathcal{B}(-A) \approx -A \wedge \mathcal{C} A$  **by** (*simp add: dual-def dual-der1 pB1 conn*)  
**lemma**  $pB5: Der-1b \mathcal{D} \Longrightarrow \forall A. \mathcal{B}(C A) \preceq (\mathcal{B} A) \vee \mathcal{B}(-A)$  **using**  $ADDI-b-def Cl-der-def MONO-ADDIb monI pB1 pB4$  **unfolding**  $conn$  **by** *auto*  
**lemma**  $pF1: \forall A. \mathcal{F} A \approx C A \leftarrow \mathcal{I} A$  **using**  $Cl-der-def Fr-der-def Int-der-def conn$  **by** *auto*  
**lemma**  $pF2: \forall A. \mathcal{F} A \approx C A \wedge \mathcal{C}(-A)$  **by** (*simp add: Cl-der-def Fr-der-def2*)  
**lemma**  $pF3: \forall A. \mathcal{F} A \approx \mathcal{B} A \vee \mathcal{B}(-A)$  **by** (*smt Br-der-def Cl-der-def dual-def dual-der1 dual-der2 pF2 conn*)  
**lemma**  $pF4: Der-1 \mathcal{D} \Longrightarrow Der-4e \mathcal{D} \Longrightarrow \forall A. \mathcal{F}(\mathcal{I} A) \preceq \mathcal{F} A$  **by** (*metis CD1 CD2 CD4a ICdual ID4 IDEM-def PC1 PC4 PC5 PD8 diff-def eq-ext' pF1*)



**lemma** *pF5*:  $Der-1 \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A. \mathcal{F}(C A) \preceq \mathcal{F} A$  **by** (*metis FD2 Fr-2-def ICdual' dual-def eq-ext' pF4*)

**lemma** *pA1*:  $\forall A. A \approx \mathcal{I} A \vee \mathcal{B} A$  **using** *Br-der-def2 Int-der-def2 conn* **by** *auto*

**lemma** *pA2*:  $\forall A. A \approx C A \leftarrow \mathcal{B}(-A)$  **using** *Cl-der-def pB4 conn* **by** *auto*

**lemma** *pC1*:  $\forall A. C A \approx A \vee \mathcal{B}(-A)$  **using** *CB-rel Cl-br-def eq-ext'* **by** *fastforce*

**lemma** *pC2*:  $\forall A. C A \approx A \vee \mathcal{F} A$  **using** *Cl-der-def Fr-der-def2 conn* **by** *auto*

**lemma** *pI1*:  $\forall A. \mathcal{I} A \approx A \leftarrow \mathcal{B} A$  **using** *pA1 pB1 conn* **by** *auto*

**lemma** *pI2*:  $\forall A. \mathcal{I} A \approx A \leftarrow \mathcal{F} A$  **using** *Br-der-def Fr-der-def pI1 conn* **by** *auto*

**lemma** *IC-imp*:  $Der-1 \mathcal{D} \implies Der-3 \mathcal{D} \implies \forall A B. \mathcal{I}(A \rightarrow B) \preceq C A \rightarrow C B$  **proof** –

**assume** *der1*:  $Der-1 \mathcal{D}$  **and** *der3*:  $Der-3 \mathcal{D}$

{ **fix** *a b*

**have**  $(a \rightarrow b) \wedge -b \rightarrow -a = \top$  **unfolding** *conn* **by** *auto*

**hence**  $\mathcal{I}((a \rightarrow b) \wedge -b \rightarrow -a) \approx \mathcal{I}(\top)$  **by** *simp*

**hence**  $\mathcal{I}((a \rightarrow b) \wedge -b \rightarrow -a) \approx \top$  **using** *der3 dNOR-def* **using** *ID3* **by** *auto*

**moreover have** *let*  $A=(a \rightarrow b) \wedge -b; B=-a$  *in*  $\mathcal{I}(A \rightarrow B) \preceq \mathcal{I}(A) \rightarrow \mathcal{I}(B)$  **using** *ID1 Int-7-def*

*PI7 der1* **by** *auto*

**ultimately have**  $\mathcal{I}((a \rightarrow b) \wedge -b) \rightarrow \mathcal{I}(-a) \approx \top$  **unfolding** *conn* **by** *simp*

**moreover have** *let*  $A=a \rightarrow b; B=-b$  *in*  $\mathcal{I}(A \wedge B) \approx \mathcal{I}(A) \wedge \mathcal{I}(B)$  **using** *ID1 MULT-def der1* **by**

*auto*

**ultimately have**  $\mathcal{I}(a \rightarrow b) \wedge \mathcal{I}(-b) \rightarrow \mathcal{I}(-a) \approx \top$  **unfolding** *conn* **by** *simp*

**moreover have**  $\forall A. \mathcal{I}(-A) \approx -C(A)$  **by** (*simp add: dual-def dual-der1 conn*)

**ultimately have**  $\mathcal{I}(a \rightarrow b) \wedge -C(b) \rightarrow -C(a) \approx \top$  **unfolding** *conn* **by** *simp*

**hence**  $\mathcal{I}(a \rightarrow b) \wedge -C(b) \preceq -C(a)$  **unfolding** *conn* **by** *simp*

**hence**  $\mathcal{I}(a \rightarrow b) \wedge C a \preceq C b$  **unfolding** *conn* **by** *metis*

} **thus** *?thesis* **unfolding** *conn* **by** *simp*

**qed**

Define some fixed-point predicates and prove some properties.

**abbreviation** *openset* (*Op*) **where**  $Op A \equiv fp \mathcal{I} A$

**abbreviation** *closedset* (*Cl*) **where**  $Cl A \equiv fp C A$

**abbreviation** *borderset* (*Br*) **where**  $Br A \equiv fp \mathcal{B} A$

**abbreviation** *frontierset* (*Fr*) **where**  $Fr A \equiv fp \mathcal{F} A$

**lemma** *Int-Open*:  $Der-1a \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A. Op(\mathcal{I} A)$  **using** *ID4a IDEM-def* **by** *blast*

**lemma** *Cl-Closed*:  $Der-1a \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A. Cl(C A)$  **using** *CD4a IDEM-def* **by** *blast*

**lemma** *Br-Border*:  $Der-1b \mathcal{D} \implies \forall A. Br(\mathcal{B} A)$  **by** (*smt Br-der-def CI1b IC1-dual PD1 conn*)

In contrast, there is no analogous fixed-point result for frontier:

**lemma**  $\mathcal{D} \mathcal{D} \implies \forall A. Fr(\mathcal{F} A)$  **nitpick oops**

**lemma** *OpCldual*:  $\forall A. Cl A \longleftrightarrow Op(-A)$  **using** *dual-def dual-der1 conn* **by** *auto*

**lemma** *ClOpdual*:  $\forall A. Op A \longleftrightarrow Cl(-A)$  **by** (*simp add: dual-def dual-der1 conn*)

**lemma** *Fr-ClBr*:  $\forall A. Fr(A) = (Cl(A) \wedge Br(A))$  **using** *join-def meet-def pB2 pC2* **by** *auto*

**lemma** *Cl-F*:  $Der-1 \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A. Cl(\mathcal{F} A)$  **using** *FD4 Fr-4-def join-def pC2* **by** *auto*

## 12.2 Further properties

The definitions and theorems below are well known in the literature (e.g. [9]). Here we uncover the minimal conditions under which they hold (taking derivative operation as primitive).

**lemma** *Cl-Bzero*:  $\forall A. Cl A \longleftrightarrow \mathcal{B}(-A) \approx \perp$  **using** *pA2 pC1* **unfolding** *conn* **by** *metis*

**lemma** *Op-Bzero*:  $\forall A. Op A \longleftrightarrow \mathcal{B} A \approx \perp$  **using** *pB1 pI1* **unfolding** *conn* **by** *metis*

**lemma** *Br-boundary*:  $\forall A. Br(A) \longleftrightarrow \mathcal{I} A \approx \perp$  **using** *Br-der-def2 Int-der-def2* **unfolding** *conn* **by** *metis*

**lemma** *Fr-nowhereDense*:  $\forall A. Fr(A) \longrightarrow \mathcal{I}(C A) \approx \perp$  **using** *Fr-ClBr Br-boundary eq-ext* **by** *metis*  
**lemma** *Cl-FB*:  $\forall A. Cl A \longleftrightarrow \mathcal{F} A \approx \mathcal{B} A$  **using** *Br-der-def2 pA2 pF1 pF3* **unfolding** *conn* **by** *metis*  
**lemma** *Op-FB*:  $\forall A. Op A \longleftrightarrow \mathcal{F} A \approx \mathcal{B}(-A)$  **using** *pA1 pA2 pF3 pI2* **unfolding** *conn* **by** *metis*  
**lemma** *Clopen-Fzero*:  $\forall A. Cl A \wedge Op A \longleftrightarrow \mathcal{F} A \approx \perp$  **using** *Cl-der-def Int-der-def Fr-der-def* **unfolding** *conn* **by** *smt*

**lemma** *Int-sup-closed*:  $Der-1b \mathcal{D} \implies supremum-closed (\lambda A. Op A)$  **by** (*smt IC1-dual ID1b Int-der-def2 PD1 sup-char diff-def*)

**lemma** *Int-meet-closed*:  $Der-1a \mathcal{D} \implies meet-closed (\lambda A. Op A)$  **by** (*metis ID1a Int-der-def MULT-b-def meet-def*)

**lemma** *Int-inf-closed*:  $Der-inf \mathcal{D} \implies infimum-closed (\lambda A. Op A)$  **by** (*simp add: fp-ID-inf-closed*)

**lemma** *Cl-inf-closed*:  $Der-1b \mathcal{D} \implies infimum-closed (\lambda A. Cl A)$  **by** (*smt Cl-der-def IC1-dual ID1b PD2 dual-der1 inf-char join-def*)

**lemma** *Cl-join-closed*:  $Der-1a \mathcal{D} \implies join-closed (\lambda A. Cl A)$  **using** *ADDI-a-def Cl-der-def join-def* **by** *fastforce*

**lemma** *Cl-sup-closed*:  $Der-inf \mathcal{D} \implies supremum-closed (\lambda A. Cl A)$  **by** (*simp add: fp-CD-sup-closed*)

**lemma** *Br-inf-closed*:  $Der-1b \mathcal{D} \implies infimum-closed (\lambda A. Br A)$  **by** (*smt Br-der-def CI1b IC1-dual PD1 inf-char diff-def*)

**lemma** *Fr-inf-closed*:  $Der-1b \mathcal{D} \implies infimum-closed (\lambda A. Fr A)$  **by** (*metis (full-types) Br-der-def Br-inf-closed Cl-der-def Cl-inf-closed Fr-der-def join-def diff-def*)

**lemma** *Br-Fr-join*:  $Der-1 \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A B. Br A \wedge Fr B \longrightarrow Br(A \vee B)$  **proof** –

**assume** *der1*:  $Der-1 \mathcal{D}$  **and** *der4*:  $Der-4e \mathcal{D}$

{ **fix**  $A B$

{ **assume** *bra*:  $Br A$  **and** *frb*:  $Fr B$

**from** *bra* **have**  $\mathcal{I} A \approx \perp$  **using** *Br-boundary* **by** *auto*

**hence**  $1: \mathcal{C}(-A) \approx \top$  **by** (*metis ICdual bottom-def compl-def dual-def eq-ext' top-def*)

**from** *frb* **have**  $\mathcal{I}(C B) \approx \perp$  **by** (*simp add: Fr-nowhereDense*)

**hence**  $2: \mathcal{C}(-(\mathcal{C} B)) \approx \top$  **by** (*metis ICdual bottom-def compl-def dual-def eq-ext' top-def*)

**from** *der1* **have**  $\mathcal{C}(-A) \leftarrow C B \preceq \mathcal{C}((-A) \leftarrow B)$  **by** (*simp add: CD1 PD4*)

**hence**  $\mathcal{C}(-A) \leftarrow C B \preceq \mathcal{C}(-(A \vee B))$  **unfolding** *conn* **by** *simp*

**hence**  $\top \leftarrow C B \preceq \mathcal{C}(-(A \vee B))$  **using**  $1$  **unfolding** *conn* **by** *simp*

**hence**  $3: -(\mathcal{C} B) \preceq \mathcal{C}(-(A \vee B))$  **unfolding** *conn* **by** *simp*

**from** *der1 der4* **have**  $4$ : *let*  $M = -(\mathcal{C} B)$ ;  $N = -(A \vee B)$  *in*  $M \preceq C N \longrightarrow C M \preceq C N$  **by** (*smt CD1b Cl-Closed PC1 PD1*)

**from**  $3\ 4$  **have**  $\mathcal{C}(-(\mathcal{C} B)) \preceq \mathcal{C}(-(A \vee B))$  **by** *simp*

**hence**  $\top \approx \mathcal{C}(-(A \vee B))$  **using**  $2$  **unfolding** *top-def* **by** *simp*

**hence**  $\perp \approx \mathcal{I}(A \vee B)$  **using** *ICdual dual-def eq-ext' conn* **by** *metis*

**hence**  $Br(A \vee B)$  **using** *Br-boundary* **by** *simp*

} **hence**  $Br A \wedge Fr B \longrightarrow Br(A \vee B)$  **by** *simp*

} **hence**  $\forall A B. Br A \wedge Fr B \longrightarrow Br(A \vee B)$  **by** *simp*

**thus** *?thesis* **by** *simp*

**qed**

**lemma** *Fr-join-closed*:  $Der-1 \mathcal{D} \implies Der-4e \mathcal{D} \implies join-closed (\lambda A. Fr A)$  **by** (*simp add: Br-Fr-join Cl-join-closed Fr-ClBr PC1*)

Introduces a predicate for indicating that two sets are disjoint and proves some properties.

**abbreviation**  $Disj A B \equiv A \wedge B \approx \perp$

**lemma** *Disj-comm*:  $\forall A B. Disj A B \longrightarrow Disj B A$  **unfolding** *conn* **by** *fastforce*

**lemma** *Disj-IF*:  $\forall A. Disj(\mathcal{I} A)(\mathcal{F} A)$  **by** (*simp add: Cl-der-def Fr-der-def2 dual-def dual-der2 conn*)

**lemma** *Disj-B*:  $\forall A. Disj(\mathcal{B} A)(\mathcal{B}(-A))$  **by** (*simp add: Br-der-def2 conn*)

**lemma** *Disj-I*:  $\forall A. Disj(\mathcal{I} A)(-A)$  **by** (*simp add: Int-der-def conn*)

**lemma** *Disj-BCI*:  $\forall A. Disj(\mathcal{B}(C A))(\mathcal{I}(-A))$  **by** (*simp add: Br-der-def2 dual-def dual-der1 conn*)

**lemma** *Disj-CBI*:  $Der-1b \mathcal{D} \implies Der-4e \mathcal{D} \implies \forall A. Disj(\mathcal{C}(\mathcal{B}(-A)))(\mathcal{I}(-A))$  **by** (*smt Br-der-def2 Der-4e-def Cl-der-def Int-der-def2 PD3 conn*)

Introduce a predicate for indicating that two sets are separated and proves some properties.

**definition**  $Sep\ A\ B \equiv Disj\ (C\ A)\ B \wedge Disj\ (C\ B)\ A$

**lemma**  $Sep\ comm: \forall A\ B. Sep\ A\ B \longrightarrow Sep\ B\ A$  **by** (*simp* *add: Sep-def*)

**lemma**  $Sep\ disj: \forall A\ B. Sep\ A\ B \longrightarrow Disj\ A\ B$  **using**  $CD2\ EXP\text{-}def\ Sep\text{-}def\ conn$  **by** *auto*

**lemma**  $Sep\ I: Der\text{-}1\ \mathcal{D} \Longrightarrow Der\text{-}4e\ \mathcal{D} \Longrightarrow \forall A. Sep\ (\mathcal{I}\ A)\ (\mathcal{I}\ (-A))$  **unfolding**  $Sep\text{-}def$  **by** (*smt*  $CD2\ CD4\ IC1\ ID1\ PC4\ PC5\ PD8\ dual\text{-}def\ dual\text{-}der1\ dual\text{-}der2\ conn$ )

**lemma**  $Sep\ sub: Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ B\ C\ D. Sep\ A\ B \wedge C \preceq A \wedge D \preceq B \longrightarrow Sep\ C\ D$  **using**  $MONO\text{-}ADD1b\ PD2\ dual\text{-}der1\ monI$  **unfolding**  $Sep\text{-}def\ conn$  **by** *metis*

**lemma**  $Sep\ Cl\text{-}diff: Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ B. Cl(A) \wedge Cl(B) \longrightarrow Sep\ (A \leftarrow B)\ (B \leftarrow A)$  **unfolding**  $Sep\text{-}def$  **using**  $CD1b\ PD1\ bottom\text{-}def\ diff\text{-}def\ meet\text{-}def$  **by** *smt*

**lemma**  $Sep\ Op\text{-}diff: Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ B. Op(A) \wedge Op(B) \longrightarrow Sep\ (A \leftarrow B)\ (B \leftarrow A)$  **proof** –  
**assume**  $der1b: Der\text{-}1b\ \mathcal{D}$

**{ fix**  $A\ B$

**from**  $der1b$  **have**  $aux: let\ M = -A ; N = -B\ in\ (Cl(M) \wedge Cl(N) \longrightarrow Sep\ (M \leftarrow N)\ (N \leftarrow M))$

**using**  $Sep\ Cl\text{-}diff$  **by** *simp*

**{ assume**  $Op(A) \wedge Op(B)$

**hence**  $Cl(-A) \wedge Cl(-B)$  **using**  $der1b\ ClOpdual$  **by** *simp*

**hence**  $Sep\ (-A \leftarrow -B)\ (-B \leftarrow -A)$  **using**  $der1b\ aux$  **unfolding** *conn* **by** *simp*

**moreover** **have**  $(-A \leftarrow -B) \approx (B \leftarrow A)$  **unfolding** *conn* **by** *auto*

**moreover** **have**  $(-B \leftarrow -A) \approx (A \leftarrow B)$  **unfolding** *conn* **by** *auto*

**ultimately** **have**  $Sep\ (B \leftarrow A)\ (A \leftarrow B)$  **unfolding** *conn* **by** *simp*

**hence**  $Sep\ (A \leftarrow B)\ (B \leftarrow A)$  **using**  $Sep\ comm$  **by** *simp*

**} hence**  $Op(A) \wedge Op(B) \longrightarrow Sep\ (A \leftarrow B)\ (B \leftarrow A)$  **by** (*rule*  $impI$ )

**} thus** *?thesis* **by** *simp*

**qed**

**lemma**  $Sep\ Cl: \forall A\ B. Cl(A) \wedge Cl(B) \wedge Disj\ A\ B \longrightarrow Sep\ A\ B$  **unfolding**  $Sep\text{-}def\ conn$  **by** *blast*

**lemma**  $Sep\ Op: Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ B. Op(A) \wedge Op(B) \wedge Disj\ A\ B \longrightarrow Sep\ A\ B$  **proof** –

**assume**  $der1b: Der\text{-}1b\ \mathcal{D}$

**{ fix**  $A\ B$

**from**  $der1b$  **have**  $aux: Op(A) \wedge Op(B) \longrightarrow Sep\ (A \leftarrow B)\ (B \leftarrow A)$  **using**  $Sep\ Op\text{-}diff$  **by** *simp*

**{ assume**  $op: Op(A) \wedge Op(B)$  **and**  $disj: Disj\ A\ B$

**hence**  $(A \leftarrow B) \approx A \wedge (B \leftarrow A) \approx B$  **unfolding** *conn* **by** *blast*

**hence**  $Sep\ A\ B$  **using**  $op\ aux$  **unfolding** *conn* **by** *simp*

**} hence**  $Op(A) \wedge Op(B) \wedge Disj\ A\ B \longrightarrow Sep\ A\ B$  **by** *simp*

**} thus** *?thesis* **by** *simp*

**qed**

**lemma**  $Der\text{-}1a\ \mathcal{D} \Longrightarrow \forall A\ B\ C. Sep\ A\ B \wedge Sep\ A\ C \longrightarrow Sep\ A\ (B \vee C)$  **using**  $ADD1\text{-}a\text{-}def\ CD1a$  **unfolding**  $Sep\text{-}def\ conn$  **by** *metis*

Verifies a neighborhood-based definition of interior.

**definition**  $nbhd\ A\ p \equiv \exists E. E \preceq A \wedge Op(E) \wedge (E\ p)$

**lemma**  $nbhd\text{-}def2: Der\text{-}1\ \mathcal{D} \Longrightarrow Der\text{-}4e\ \mathcal{D} \Longrightarrow \forall A\ p. (nbhd\ A\ p) = (\mathcal{I}\ A\ p)$  **unfolding**  $nbhd\text{-}def$  **by** (*smt*  $Int\text{-}Open\ MONO\text{-}def\ PC1\ monI\ pI2\ conn$ )

**lemma**  $I\text{-}def'\text{-}lr': \forall A\ p. (\mathcal{I}\ A\ p) \longrightarrow (\exists E. (\mathcal{I}\ E\ p) \wedge E \preceq A)$  **by** *blast*

**lemma**  $I\text{-}def'\text{-}rl': Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ p. (\mathcal{I}\ A\ p) \longleftarrow (\exists E. (\mathcal{I}\ E\ p) \wedge E \preceq A)$  **using**  $MONO\text{-}def\ monI$  **by** *metis*

**lemma**  $I\text{-}def': Der\text{-}1b\ \mathcal{D} \Longrightarrow \forall A\ p. (\mathcal{I}\ A\ p) \longleftrightarrow (\exists E. (\mathcal{I}\ E\ p) \wedge E \preceq A)$  **using**  $MONO\text{-}def\ monI$  **by** *metis*

Explore the Barcan and converse Barcan formulas.

**lemma**  $Barcan\text{-}I: Der\text{-}inf\ \mathcal{D} \Longrightarrow \forall P. (\forall x. \mathcal{I}(P\ x)) \preceq \mathcal{I}(\forall x. P\ x)$  **using**  $ID\text{-}inf\ Barcan1$  **by** *auto*

**lemma**  $Barcan\text{-}C: Der\text{-}inf\ \mathcal{D} \Longrightarrow \forall P. \mathcal{C}(\exists x. P\ x) \preceq (\exists x. \mathcal{C}(P\ x))$  **using**  $CD\text{-}inf\ Barcan2$  **by** *metis*

**lemma** *CBarcan-I*:  $Der-1b \mathcal{D} \implies \forall P. \mathcal{I}(\forall x. P x) \preceq (\forall x. \mathcal{I}(P x))$  **by** (*metis (mono-tags, lifting) MONO-def monI*)  
**lemma** *CBarcan-C*:  $Der-1b \mathcal{D} \implies \forall P. (\exists x. \mathcal{C}(P x)) \preceq \mathcal{C}(\exists x. P x)$  **by** (*metis (mono-tags, lifting) MONO-def monC*)

**end**

**theory** *ex-LFUs*

**imports** *topo-derivative-algebra sse-operation-negative*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 13 Example application: Logics of Formal Undeterminedness (LFUs)

The LFUs [10] [4] are a family of paracomplete logics featuring a 'determinedness' operator  $\circ$  that can be used to recover some classical properties of negation (in particular TND). LFUs behave in a sense dually to LFIs. Both can be semantically embedded as extensions of Boolean algebras. Here we show how to semantically embed LFUs as derivative algebras.

(We rename (classical) meta-logical negation to avoid terminological confusion)

**abbreviation** *cneg*:: $bool \implies bool$  ( $\sim$ - [40]40) **where**  $\sim \varphi \equiv \neg \varphi$

Logical validity can be defined as truth in all worlds/points (i.e. as denoting the top element)

**abbreviation** *gtrue*:: $\sigma \implies bool$  ( $\vdash$  -) **where**  $\vdash A \equiv \forall w. A w$

**lemma** *gtrue-def*:  $\vdash A \equiv A \approx \top$  **by** (*simp add: top-def*)

As for LFIs, we focus on the local (truth-degree preserving) notion of logical consequence.

**abbreviation** *conseq-local1*:: $\sigma \implies \sigma \implies bool$  ( $\vdash$  -) **where**  $[A \vdash B] \equiv A \preceq B$

**abbreviation** *conseq-local2*:: $\sigma \implies \sigma \implies \sigma \implies bool$  ( $\vdash$ , -) **where**  $[A_1, A_2 \vdash B] \equiv A_1 \wedge A_2 \preceq B$

**abbreviation** *conseq-local12*:: $\sigma \implies \sigma \implies \sigma \implies bool$  ( $\vdash$  -, -) **where**  $[A \vdash B_1, B_2] \equiv A \preceq B_1 \vee B_2$

For LFUs we use the interior-based negation previously defined (taking derivative as primitive).

**definition** *ineg*:: $\sigma \implies \sigma$  ( $\neg$ ) **where**  $\neg A \equiv \mathcal{I}(-A)$

**declare** *ineg-def*[*conn*]

In terms of the derivative operator the negation looks as follows:

**lemma** *ineg-prop*:  $\neg A \approx -(\mathcal{D} A) \leftarrow A$  **using** *Cl-der-def IB-rel Int-br-def eq-ext' pB4 conn* **by** *fastforce*

This negation is of course paracomplete.

**lemma**  $\vdash a \vee \neg a$  **nitpick oops**

We define two pairs of in/determinedness operators and show how they relate to each other.

**abbreviation** *op-det*:: $\sigma \implies \sigma$  ( $\circ$ - [57]58) **where**  $\circ A \equiv \mathcal{B}^d A$

**abbreviation** *op-ind*:: $\sigma \implies \sigma$  ( $\cdot$ - [57]58) **where**  $\cdot A \equiv -\circ A$

**lemma** *op-det-def*:  $\circ a \approx a \vee \neg a$  **by** (*simp add: compl-def diff-def dual-def ineg-def join-def pB1*)

**lemma** *Prop1*:  $\circ A \approx \mathcal{C}^{fp} A$  **by** (*metis dimp-def eq-ext' fp3*)

**lemma** *Prop2*:  $Op A \longleftrightarrow \circ -A \approx \top$  **by** (*metis dual-def dual-symm pB1 pI1 top-def compl-def diff-def*)

**lemma** *Prop3*:  $Op A \longleftrightarrow \cdot -A \approx \perp$  **by** (*metis Op-Bzero dual-def dual-symm*)

**lemma** *Prop4*:  $Cl A \longleftrightarrow \circ A \approx \top$  **by** (*metis Prop1 dimp-def top-def*)

**lemma** *Prop5*:  $Cl A \longleftrightarrow \cdot A \approx \perp$  **by** (*simp add: Prop4 bottom-def compl-def top-def*)

Analogously as for LFIs, LFUs provide means for recovering the principle of excluded middle.

**lemma**  $[\Gamma \vdash \cdot a, a \vee \neg a]$  **using** *IB-rel Int-br-def compl-def diff-def dual-def eq-ext' ineg-def join-def* **by** *fastforce*

**lemma**  $[\Gamma, \circ a \vdash a \vee \neg a]$  **using** *dual-def pB1 unfolding conn* **by** *auto*

**lemma**  $TNDm(\neg)$  **nitpick oops**

**lemma**  $ECQ(\neg)$  **by** (*simp add: ECQ-def bottom-def diff-def ineg-prop meet-def*)

**lemma**  $Der-3 \mathcal{D} \implies LNC(\neg)$  **using** *ineg-prop ECQ-def ID3 LNC-def dNOR-def* **unfolding** *conn* **by** *auto*

**lemma**  $\mathfrak{D} \mathcal{D} \implies DNI(\neg)$  **nitpick oops**

**lemma**  $\mathfrak{D} \mathcal{D} \implies DNE(\neg)$  **nitpick oops**

**lemma**  $Der-1b \mathcal{D} \implies CoPw(\neg)$  **by** (*smt CoPw-def MONO-ADDIb PD1 compl-def ineg-def monI*)

**lemma**  $\mathfrak{D} \mathcal{D} \implies CoP1(\neg)$  **nitpick oops**

**lemma**  $\mathfrak{D} \mathcal{D} \implies CoP2(\neg)$  **nitpick oops**

**lemma**  $\mathfrak{D} \mathcal{D} \implies CoP3(\neg)$  **nitpick oops**

**lemma**  $\mathfrak{D} \mathcal{D} \implies DM3(\neg)$  **nitpick oops**

**lemma**  $Der-1a \mathcal{D} \implies DM4(\neg)$  **unfolding** *Defs* **using** *ADDI-a-def ineg-prop compl-def diff-def join-def meet-def* **by** *auto*

**lemma**  $Der-3 \mathcal{D} \implies nNor(\neg)$  **by** (*simp add: NOR-def ineg-prop nNor-def bottom-def compl-def diff-def top-def*)

**lemma**  $nDNor(\neg)$  **by** (*simp add: bottom-def diff-def ineg-prop nDNor-def top-def*)

**lemma**  $Der-1b \mathcal{D} \implies MT0(\neg)$  **unfolding** *Defs* **by** (*metis (mono-tags, hide-lams) CD1b Disj-I OpCld-ual PD1 bottom-def compl-def ineg-def meet-def top-def*)

**lemma**  $Der-1b \mathcal{D} \implies Der-3 \mathcal{D} \implies MT1(\neg)$  **unfolding** *Defs* **by** (*metis (full-types) NOR-def PD1 bottom-def compl-def diff-def ineg-prop top-def*)

**lemma**  $\mathfrak{D} \mathcal{D} \implies MT2(\neg)$  **nitpick oops**

**lemma**  $\mathfrak{D} \mathcal{D} \implies MT3(\neg)$  **nitpick oops**

We show how all local contraposition variants (lCoP) can be recovered using the determinedness operator. Observe that we can recover in the same way other (weaker) properties: CoP, MT and DNI/DNE (local & global).

**lemma**  $\mathfrak{D} \mathcal{D} \implies lCoPw(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *det-lcop1*:  $[\circ a, a \rightarrow b \vdash \neg b \rightarrow \neg a]$  **using** *dual-def pI1 conn* **by** *auto*

**lemma**  $\mathfrak{D} \mathcal{D} \implies lCoP1(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *det-lcop2*:  $[\circ a, a \rightarrow \neg b \vdash b \rightarrow \neg a]$  **using** *dual-def pI1 conn* **by** *auto*

**lemma**  $\mathfrak{D} \mathcal{D} \implies lCoP2(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *det-lcop3*:  $[\circ a, \neg a \rightarrow b \vdash \neg b \rightarrow a]$  **using** *dual-def pI1 conn* **by** *auto*

**lemma**  $\mathfrak{D} \mathcal{D} \implies lCoP3(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *det-lcop4*:  $[\circ a, \neg a \rightarrow \neg b \vdash b \rightarrow a]$  **using** *dual-def pI1 conn* **by** *auto*

Disjunctive syllogism (DS).

**lemma**  $DS1(\rightarrow)(\neg)$  **by** (*simp add: DS1-def diff-def impl-def ineg-prop join-def*)

**lemma**  $\mathfrak{D} \mathcal{D} \implies DS2(\rightarrow)(\neg)$  **nitpick oops**

**lemma** *det-ds2*:  $[\circ a, \neg a \rightarrow b \vdash a \vee b]$  **using** *pB1 dual-def conn* **by** *auto*

**end**

**theory** *topo-border-algebra*

**imports** *topo-operators-basic*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 14 Border algebra

We define a border algebra in an analogous fashion to the well-known closure/interior algebras. We also verify a few interesting properties.

Declares a primitive (unconstrained) border operation and defines others from it.

**consts**  $\mathcal{B}::\sigma\Rightarrow\sigma$   
**abbreviation**  $\mathcal{I} \equiv \mathcal{I}_B \mathcal{B}$  — interior  
**abbreviation**  $\mathcal{C} \equiv \mathcal{C}_B \mathcal{B}$  — closure  
**abbreviation**  $\mathcal{F} \equiv \mathcal{F}_B \mathcal{B}$  — frontier

### 14.1 Basic properties

Verifies minimal conditions under which operators resulting from conversion functions coincide.

**lemma** *ICdual*:  $\mathcal{I} \equiv \mathcal{C}^d$  **by** (*simp add: Cl-br-def Int-br-def dual-def equal-op-def conn*)  
**lemma** *ICdual'*:  $\mathcal{C} \equiv \mathcal{I}^d$  **by** (*simp add: Cl-br-def Int-br-def dual-def equal-op-def conn*)  
**lemma** *FI-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{F} \equiv \mathcal{F}_I \mathcal{I}$  **using** *Fr-br-def Fr-int-def Int-br-def equal-op-def* **by** (*smt Br-5b-def PB5b dual-def conn*)  
**lemma** *IF-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{I} \equiv \mathcal{I}_F \mathcal{F}$  **using** *Br-5b-def Fr-br-def Int-br-def Int-fr-def PB5b unfolding equal-op-def conn* **by** *fastforce*  
**lemma** *FC-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{F} \equiv \mathcal{F}_C \mathcal{C}$  **using** *Br-5b-def Cl-br-def Fr-br-def Fr-cl-def PB5b unfolding equal-op-def conn* **by** *fastforce*  
**lemma** *CF-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{C} \equiv \mathcal{C}_F \mathcal{F}$  **using** *Br-5b-def Cl-br-def Cl-fr-def Fr-br-def PB5b unfolding equal-op-def conn* **by** *fastforce*  
**lemma** *BI-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{B} \equiv \mathcal{B}_I \mathcal{I}$  **using** *Br-5b-def Br-int-def Int-br-def PB5b diff-def equal-op-def* **by** *fastforce*  
**lemma** *BC-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{B} \equiv \mathcal{B}_C \mathcal{C}$  **using** *BI-BC-rel BI-rel ICdual' eq-ext'* **by** *fastforce*  
**lemma** *BF-rel*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{B} \equiv \mathcal{B}_F \mathcal{F}$  **by** (*smt BI-rel Br-fr-def Br-int-def IF-rel Int-fr-def diff-def equal-op-def meet-def*)

Fixed-point and other operators are interestingly related.

**lemma** *fp1*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{I}^{fp} \equiv \mathcal{B}^c$  **using** *Br-5b-def Int-br-def PB5b unfolding equal-op-def conn* **by** *fastforce*  
**lemma** *fp2*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{B}^{fp} \equiv \mathcal{I}^c$  **using** *Br-5b-def Int-br-def PB5b conn equal-op-def* **by** *fastforce*  
**lemma** *fp3*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{C}^{fp} \equiv \mathcal{B}^d$  **using** *Br-5c-def Cl-br-def PB5c dual-def unfolding equal-op-def conn* **by** *fastforce*  
**lemma** *fp4*:  $Br-1 \mathcal{B} \Longrightarrow (\mathcal{B}^d)^{fp} \equiv \mathcal{C}$  **by** (*smt dimp-def equal-op-def fp3*)  
**lemma** *fp5*:  $Br-1 \mathcal{B} \Longrightarrow \mathcal{F}^{fp} \equiv \mathcal{B} \sqcup (\mathcal{C}^c)$  **by** (*smt Br-5b-def Cl-br-def Fr-br-def PB5b equal-op-def conn*)

Define some fixed-point predicates and prove some properties.

**abbreviation** *openset* (*Op*) **where**  $Op A \equiv fp \mathcal{I} A$   
**abbreviation** *closedset* (*Cl*) **where**  $Cl A \equiv fp \mathcal{C} A$   
**abbreviation** *borderset* (*Br*) **where**  $Br A \equiv fp \mathcal{B} A$   
**abbreviation** *frontierset* (*Fr*) **where**  $Fr A \equiv fp \mathcal{F} A$

**lemma** *Int-Open*:  $Br-1 \mathcal{B} \Longrightarrow Br-3 \mathcal{B} \Longrightarrow \forall A. Op(\mathcal{I} A)$  **using** *IB4 IDEM-def* **by** *blast*  
**lemma** *Cl-Closed*:  $Br-1 \mathcal{B} \Longrightarrow Br-3 \mathcal{B} \Longrightarrow \forall A. Cl(\mathcal{C} A)$  **using** *CB4 IDEM-def* **by** *blast*  
**lemma** *Br-Border*:  $Br-1 \mathcal{B} \Longrightarrow \forall A. Br(\mathcal{B} A)$  **using** *IDEM-def PB6* **by** *blast*

In contrast, there is no analogous fixed-point result for frontier:

**lemma**  $\mathfrak{B} \mathcal{B} \Longrightarrow \forall A. Fr(\mathcal{F} A)$  **nitpick oops**

**lemma** *OpClDual*:  $\forall A. Cl A \longleftrightarrow Op(-A)$  **using** *Cl-br-def Int-br-def conn* **by** *auto*

```

lemma ClOpdual:  $\forall A. Op\ A \longleftrightarrow Cl(-A)$  using Cl-br-def Int-br-def conn by auto
lemma Fr-ClBr:  $Br-1\ \mathcal{B} \implies \forall A. Fr(A) = (Cl(A) \wedge Br(A))$  by (metis BF-rel Br-fr-def CF-rel Cl-fr-def eq-ext' join-def meet-def)
lemma Cl-F:  $Br-1\ \mathcal{B} \implies Br-3\ \mathcal{B} \implies \forall A. Cl(\mathcal{F}\ A)$  by (metis CF-rel Cl-fr-def FB4 Fr-4-def eq-ext' join-def)

end
theory topo-closure-algebra
  imports topo-operators-basic
begin
nitpick-params[assms=true, user-axioms=true, show-all, expect=genuine, format=3]

```

## 15 Closure algebra

We define a topological Boolean algebra with a primitive closure operator and verify a few properties.

Declares a primitive (unconstrained) closure operation and defines others from it.

```

consts C:: $\sigma \Rightarrow \sigma$ 
abbreviation  $\mathcal{I} \equiv C^d$  — interior
abbreviation  $\mathcal{B} \equiv \mathcal{B}_C\ C$  — border
abbreviation  $\mathcal{F} \equiv \mathcal{F}_C\ C$  — frontier

```

### 15.1 Basic properties

Verifies minimal conditions under which operators resulting from conversion functions coincide.

```

lemma ICdual':  $C \equiv \mathcal{I}^d$  using dual-symm equal-op-def by auto
lemma IB-rel:  $Cl-2\ C \implies \mathcal{I} \equiv \mathcal{I}_B\ \mathcal{B}$  using Br-cl-def EXP-dual1 Int-br-def compl-def dEXP-def diff-def dual-def equal-op-def meet-def by fastforce
lemma IF-rel:  $Cl-2\ C \implies \mathcal{I} \equiv \mathcal{I}_F\ \mathcal{F}$  by (smt EXP-def Fr-cl-def Int-fr-def compl-def diff-def dual-def equal-op-def meet-def)
lemma CB-rel:  $Cl-2\ C \implies C \equiv \mathcal{C}_B\ \mathcal{B}$  by (smt Cl-br-def EXP-def IB-rel Int-br-def compl-def diff-def dual-def dual-symm eq-ext' equal-op-def join-def)
lemma CF-rel:  $Cl-2\ C \implies C \equiv \mathcal{C}_F\ \mathcal{F}$  by (smt Br-cl-def CB-rel Cl-br-def Cl-fr-def Fr-cl-def equal-op-def join-def meet-def)
lemma BI-rel:  $\mathcal{B} \equiv \mathcal{B}_I\ \mathcal{I}$  using BI-BC-rel dual-symm equal-op-def by metis
lemma BF-rel:  $Cl-2\ C \implies \mathcal{B} \equiv \mathcal{B}_F\ \mathcal{F}$  by (smt Br-cl-def Br-fr-def EXP-def Fr-cl-def equal-op-def meet-def)
lemma FI-rel:  $\mathcal{F} \equiv \mathcal{F}_I\ \mathcal{I}$  by (metis FI2 Fr-2-def Fr-cl-def Fr-int-def ICdual' dual-def eq-ext' equal-op-def)
lemma FB-rel:  $Cl-2\ C \implies \mathcal{F} \equiv \mathcal{F}_B\ \mathcal{B}$  by (smt BF-rel Br-fr-def CB-rel Cl-br-def Fr-br-def Fr-cl-def equal-op-def join-def meet-def)

```

Fixed-point and other operators are interestingly related.

```

lemma fp1:  $Cl-2\ C \implies \mathcal{I}^{fp} \equiv \mathcal{B}^c$  by (smt BI-rel Br-int-def IB-rel Int-br-def compl-def diff-def dimp-def equal-op-def)
lemma fp2:  $Cl-2\ C \implies \mathcal{B}^{fp} \equiv \mathcal{I}^c$  using fp1 by (smt compl-def dimp-def equal-op-def)
lemma fp3:  $Cl-2\ C \implies \mathcal{C}^{fp} \equiv \mathcal{B}^d$  by (smt BI-rel Br-int-def CB-rel Cl-br-def compl-def diff-def dimp-def dual-def equal-op-def join-def)
lemma fp4:  $Cl-2\ C \implies (\mathcal{B}^d)^{fp} \equiv C$  by (smt dimp-def equal-op-def fp3)
lemma fp5:  $Cl-2\ C \implies \mathcal{F}^{fp} \equiv \mathcal{B} \sqcup (C^c)$  by (smt Br-cl-def CF-rel Cl-fr-def FC2 Fr-2-def compl-def dimp-def eq-ext' equal-op-def join-def meet-def)

```

Define some fixed-point predicates and prove some properties.

**abbreviation** *openset* (*Op*) **where**  $Op\ A \equiv fp\ \mathcal{I}\ A$   
**abbreviation** *closedset* (*Cl*) **where**  $Cl\ A \equiv fp\ \mathcal{C}\ A$   
**abbreviation** *borderset* (*Br*) **where**  $Br\ A \equiv fp\ \mathcal{B}\ A$   
**abbreviation** *frontierset* (*Fr*) **where**  $Fr\ A \equiv fp\ \mathcal{F}\ A$

**lemma** *Int-Open*:  $Cl\text{-}4\ \mathcal{C} \implies \forall A. Op(\mathcal{I}\ A)$  **using** *IC4 IDEM-def* **by** *blast*

**lemma** *Cl-Closed*:  $Cl\text{-}4\ \mathcal{C} \implies \forall A. Cl(\mathcal{C}\ A)$  **by** (*simp add: IDEM-def*)

**lemma** *Br-Border*:  $Cl\text{-}1b\ \mathcal{C} \implies \forall A. Br(\mathcal{B}\ A)$  **by** (*metis BI-rel Br-cl-def Br-int-def CI1b MULT-a-def diff-def eq-ext' meet-def*)

In contrast, there is no analogous fixed-point result for frontier:

**lemma**  $\mathcal{C}\ \mathcal{C} \implies \forall A. Fr(\mathcal{F}\ A)$  **nitpick oops**

**lemma** *OpClDual*:  $\forall A. Cl\ A \longleftrightarrow Op(-A)$  **by** (*simp add: compl-def dual-def*)

**lemma** *ClOpDual*:  $\forall A. Op\ A \longleftrightarrow Cl(-A)$  **by** (*simp add: fp-d*)

**lemma** *Fr-ClBr*:  $Cl\text{-}2\ \mathcal{C} \implies \forall A. Fr(A) = (Cl(A) \wedge Br(A))$  **using** *BF-rel* **by** (*metis Br-fr-def CF-rel Cl-fr-def eq-ext' join-def meet-def*)

**lemma** *Cl-F*:  $Cl\text{-}1b\ \mathcal{C} \implies Cl\text{-}2\ \mathcal{C} \implies Cl\text{-}4\ \mathcal{C} \implies \forall A. Cl(\mathcal{F}\ A)$  **using** *Cl-8-def Fr-cl-def PC8* **by** *auto*

**end**

**theory** *topo-interior-algebra*

**imports** *topo-operators-basic*

**begin**

**nitpick-params**[*assms=true, user-axioms=true, show-all, expect=genuine, format=3*]

## 16 Interior algebra

We define a topological Boolean algebra taking the interior operator as primitive and verify some properties.

Declares a primitive (unconstrained) interior operation and defines others from it.

**consts**  $\mathcal{I}::\sigma \Rightarrow \sigma$

**abbreviation**  $\mathcal{C} \equiv \mathcal{I}^d$  — closure

**abbreviation**  $\mathcal{B} \equiv \mathcal{B}_I\ \mathcal{I}$  — border

**abbreviation**  $\mathcal{F} \equiv \mathcal{F}_I\ \mathcal{I}$  — frontier

### 16.1 Basic properties

Verifies minimal conditions under which operators resulting from conversion functions coincide.

**lemma** *ICdual*:  $\mathcal{I} \equiv \mathcal{C}^d$  **using** *dual-symm equal-op-def* **by** *auto*

**lemma** *IB-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{I} \equiv \mathcal{I}_B\ \mathcal{B}$  **by** (*smt Br-int-def Int-br-def dEXP-def diff-def equal-op-def*)

**lemma** *IF-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{I} \equiv \mathcal{I}_F\ \mathcal{F}$  **using** *EXP-def EXP-dual2 Fr-int-def IB-rel Int-br-def Int-fr-def compl-def diff-def equal-op-def meet-def* **by** *fastforce*

**lemma** *CB-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{C} \equiv \mathcal{C}_B\ \mathcal{B}$  **using** *Br-int-def Cl-br-def EXP-def EXP-dual2 compl-def diff-def dual-def equal-op-def join-def* **by** *fastforce*

**lemma** *CF-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{C} \equiv \mathcal{C}_F\ \mathcal{F}$  **by** (*smt Cl-fr-def EXP-def EXP-dual2 Fr-int-def compl-def dEXP-def equal-op-def join-def meet-def*)

**lemma** *BC-rel*:  $\mathcal{B} \equiv \mathcal{B}_C\ \mathcal{C}$  **by** (*simp add: BI-BC-rel equal-op-def*)

**lemma** *BF-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{B} \equiv \mathcal{B}_F\ \mathcal{F}$  **by** (*smt Br-fr-def Br-int-def IF-rel Int-fr-def diff-def equal-op-def meet-def*)

**lemma** *FC-rel*:  $\mathcal{F} \equiv \mathcal{F}_C\ \mathcal{C}$  **using** *Fr-cl-def Fr-int-def compl-def dual-def equal-op-def meet-def* **by** *fastforce*

**lemma** *FB-rel*:  $Int\text{-}2\ \mathcal{I} \implies \mathcal{F} \equiv \mathcal{F}_B\ \mathcal{B}$  **unfolding** *Fr-br-def* **by** (*smt BC-rel Br-cl-def CB-rel Cl-br-def FC-rel Fr-cl-def equal-op-def join-def meet-def*)



Fixed-point and other operators are interestingly related.

**lemma** *fp1*:  $\text{Int-2 } \mathcal{I} \Longrightarrow \mathcal{I}^{fp} \equiv \mathcal{B}^c$  **by** (*smt Br-int-def IB-rel Int-br-def compl-def diff-def dimp-def equal-op-def*)

**lemma** *fp2*:  $\text{Int-2 } \mathcal{I} \Longrightarrow \mathcal{B}^{fp} \equiv \mathcal{I}^c$  **using** *fp1 unfolding compl-def dimp-def equal-op-def* **by** *smt*

**lemma** *fp3*:  $\text{Int-2 } \mathcal{I} \Longrightarrow \mathcal{C}^{fp} \equiv \mathcal{B}^d$  **by** (*metis (no-types) dual-comp eq-ext' fp2 ofp-c ofp-d ofp-invol*)

**lemma** *fp4*:  $\text{Int-2 } \mathcal{I} \Longrightarrow (\mathcal{B}^d)^{fp} \equiv \mathcal{C}$  **by** (*smt dimp-def equal-op-def fp3*)

**lemma** *fp5*:  $\text{Int-2 } \mathcal{I} \Longrightarrow \mathcal{F}^{fp} \equiv \mathcal{B} \sqcup (\mathcal{C}^c)$  **by** (*smt BC-rel Br-cl-def CF-rel Cl-fr-def FI2 Fr-2-def compl-def dimp-def eq-ext' equal-op-def join-def meet-def*)

Define some fixed-point predicates and prove some properties.

**abbreviation** *openset* (*Op*) **where**  $Op\ A \equiv fp\ \mathcal{I}\ A$

**abbreviation** *closedset* (*Cl*) **where**  $Cl\ A \equiv fp\ \mathcal{C}\ A$

**abbreviation** *borderset* (*Br*) **where**  $Br\ A \equiv fp\ \mathcal{B}\ A$

**abbreviation** *frontierset* (*Fr*) **where**  $Fr\ A \equiv fp\ \mathcal{F}\ A$

**lemma** *Int-Open*:  $\text{Int-4 } \mathcal{I} \Longrightarrow \forall A. Op(\mathcal{I}\ A)$  **by** (*simp add: IDEM-def*)

**lemma** *Cl-Closed*:  $\text{Int-4 } \mathcal{I} \Longrightarrow \forall A. Cl(\mathcal{C}\ A)$  **using** *IC4 IDEM-def* **by** *blast*

**lemma** *Br-Border*:  $\text{Int-1a } \mathcal{I} \Longrightarrow \forall A. Br(\mathcal{B}\ A)$  **by** (*metis Br-int-def MONO-MULTa MONO-def diff-def*)

In contrast, there is no analogous fixed-point result for frontier:

**lemma**  $\exists \mathcal{I} \Longrightarrow \forall A. Fr(\mathcal{F}\ A)$  **nitpick oops**

**lemma** *OpClDual*:  $\forall A. Cl\ A \longleftrightarrow Op(-A)$  **by** (*simp add: fp-d*)

**lemma** *ClOpDual*:  $\forall A. Op\ A \longleftrightarrow Cl(-A)$  **by** (*simp add: compl-def dual-def*)

**lemma** *Fr-ClBr*:  $\text{Int-2 } \mathcal{I} \Longrightarrow \forall A. Fr(A) = (Cl(A) \wedge Br(A))$  **using** *BF-rel Br-fr-def CF-rel Cl-fr-def eq-ext' join-def meet-def* **by** *fastforce*

**lemma** *Cl-F*:  $\text{Int-1a } \mathcal{I} \Longrightarrow \text{Int-2 } \mathcal{I} \Longrightarrow \text{Int-4 } \mathcal{I} \Longrightarrow \forall A. Cl(\mathcal{F}\ A)$  **by** (*metis CF-rel Cl-fr-def FI4 Fr-4-def eq-ext' join-def*)

**end**

## References

- [1] C. Benzmüller. Universal (meta-)logical reasoning: Recent successes. *Science of Computer Programming*, 172:48–62, 2019. Preprint: <http://doi.org/10.13140/RG.2.2.11039.61609/2>.
- [2] C. Benzmüller and L. Paulson. Quantified multimodal logics in simple type theory. *Logica Universalis (Special Issue on Multimodal Logics)*, 7(1):7–20, 2013. Preprint: <http://christoph-benzmueller.de/papers/J23.pdf>.
- [3] W. Carnielli, M. E. Coniglio, and D. Fuenmayor. Logics of formal inconsistency enriched with replacement: an algebraic and modal account. *arXiv*, math.LO(2003.09522), 2020.
- [4] W. Carnielli, M. E. Coniglio, and J. Marcos. Logics of formal inconsistency. In *Handbook of philosophical logic*, pages 1–93. Springer, 2007.
- [5] L. Esakia. Intuitionistic logic and modality via topology. *Annals of Pure and Applied Logic*, 127(1-3):155–170, 2004.
- [6] F. Hausdorff. *Grundzüge der Mengenlehre*, volume 7. von Veit, 1914.
- [7] I. Johansson. Der Minimalkalkül, ein reduzierter intuitionistischer Formalismus. *Compositio mathematica*, 4:119–136, 1937.

- [8] K. Kuratowski. Sur l'opération  $\bar{A}$  de l'analysis situs. *Fundamenta Mathematicae*, 3(1):182–199, 1922.
- [9] K. Kuratowski. *Topology: Volume I*, volume 1. Elsevier, 2014.
- [10] J. Marcos. Nearly every normal modal logic is paranormal. *Logique et Analyse*, 48(189/192):279–300, 2005.
- [11] J. C. C. McKinsey and A. Tarski. The algebra of topology. *Annals of mathematics*, pages 141–191, 1944.
- [12] M. Zarycki. Quelques notions fondamentales de l'analysis situs au point de vue de l'algèbre de la logique. *Fundamenta Mathematicae*, 9(1):3–15, 1927. English translation by Mark Bowron: <https://www.researchgate.net/scientific-contributions/Miron-Zarycki-2016157096>.
- [13] M. Zarycki. Allgemeine Eigenschaften der cantorschen Kohärenzen. *Transactions of the American Mathematical Society*, 30(3):498–506, 1928. English translation by Mark Bowron: <https://www.researchgate.net/scientific-contributions/Miron-Zarycki-2016157096>.
- [14] M. Zarycki. Some properties of the derived set operation in abstract spaces. *Nauk. Zap. Ser. Fiz.-Mat.*, 5:22–33, 1947. English translation by Mark Bowron: <https://www.researchgate.net/scientific-contributions/Miron-Zarycki-2016157096>.