UNIVERSITÄT AUGSBURG

An Algebra for Abstract Interfaces

Peter Höfner Bernhard Möller

Report 2010-02 April 2010

INSTITUT FÜR INFORMATIK
D-86135 AUGSBURG

An Algebra for Abstract Interfaces

Peter Höfner
Institut für Informatik, Universität Augsburg
D-86135 Augsburg, Germany
hoefner@informatik.uni-augsburg.de

Bernhard Möller
Institut für Informatik, Universität Augsburg
D-86135 Augsburg, Germany
h.dang@informatik.uni-augsburg.de

Abstract

ABSTRACT

1 Introduction

In [4] we introduced an algebra for abstract interfaces. In this report, we look at this algebra in detail. In particular, we give some examples, derive properties and develop an equivalent characterisation.

2 Restriction and Complement

We now assume the set C of code fragments to be a semilattice L [1] with least element 0. The operator $\sqcup$ denotes the supremum in L and can be viewed as forming the union of two code fragments. The semilattice order is denoted by $\sqsubseteq$ and corresponds to the inclusion order on sets. The least element 0 can be thought of as the empty code fragment. The set T of abstract interfaces is abstracted to a sublattice N of L with $0 \in N$.

Definition 2.1 (restriction and complement)**.** The *restriction* $\lfloor : L \times N \to L$ and the *complement* $- : L \times N \to L$ have to satisfy the following axioms:

$$
\begin{array}{rcll@{\qquad}rcll}
(a \sqcup b)\lfloor p & = & (a\lfloor p) \sqcup (b\lfloor p) & (1) & a & = & (a\lfloor p) \sqcup (a-p) & (6)\\
a\lfloor(p \sqcup q) & = & (a\lfloor p) \sqcup (a\lfloor q) & (2) & a\lfloor(q-p) & = & (a-p)\lfloor q = (a\lfloor q)-p & (7)\\
(a \sqcup b) - p & = & (a-p) \sqcup (b-p) & (3) & p-p & = & 0 & (8)\\
a-(p \sqcup q) & = & (a-p)-q & (4) & p\lfloor q & \sqsubseteq & q & (9)\\
a\lfloor 0 & = & 0 & (5) & & & &
\end{array}
$$

where $a,b \in L$, $p,q \in N$. Moreover we assume that N is closed under $\lfloor$ and $-$, i.e.,if $p,q \in N$ then also $p\lfloor q, p-q \in N$.

To illustrate the definition and to show that there are models that satisfy the axiom, we now present some examples.

Example 2.2.

- Finite examples are generated by Mace4 [5].

Dimension 2 There is only one model of dimension 2:

ai	0	1
	1	1

$\leq$	0	1
0	1	1
1	0	1

$\sqcup$	0	1
0	0	1
1	1	1

$-$	0	1
0	0	0
1	1	0

$\lfloor$	0	1
0	0	0
1	0	1

Dimension 3 There is only one model of dimension 2:

ai	0	1
	1	1

$\leq$	0	1
0	1	1
1	0	1

$\sqcup$	0	1
0	0	1
1	1	1

$-$	0	1
0	0	0
1	1	0

$\lfloor$	0	1
0	0	0
1	0	1

□

We list a couple of properties for the newly introduced operators. For example Equations (6) and (8) imply $p \lfloor p = p$. We only present the most important properties; many more can be found in [4]. Proofs by hand are given in Appendix **??**. Note that we can employ some useful properties induced by the semilattice; examples are $a \sqsubseteq b \Leftrightarrow a \sqcup b = b$, $a \sqsubseteq 0 \Rightarrow a = 0$ and

$$a \sqcup b \sqsubseteq c \Leftrightarrow a \sqsubseteq c \wedge b \sqsubseteq c \tag{10}$$

A further one is that each equation can be split into inequalities, i.e., $a = b \Leftrightarrow (a \sqsubseteq b \wedge b \sqsubseteq a)$.

Lemma 2.3. *Assume arbitrary code fragments $a, b \in L$ and $p, q \in N$.*

(a) $a \lfloor p \sqsubseteq a$. *In particular* $0 \lfloor p = 0$.

(b) $a - p \sqsubseteq a$. *In particular* $0 - p = 0$. *Moreover* $a - 0 = a$.

(c) Restriction is isotone in both arguments, i.e., $a \sqsubseteq b \Rightarrow a \lfloor p \sqsubseteq b \lfloor p$ *and* $p \sqsubseteq q \Rightarrow a \lfloor p \sqsubseteq a \lfloor q$.

(d) The operator $-$ is isotone in its left and antitone in its right argument, i.e., $a \sqsubseteq b \Rightarrow a - p \sqsubseteq b - p$ *and* $p \sqsubseteq q \Rightarrow a - q \sqsubseteq a - p$,

(e) Restriction is quasi-associative, i.e., $a \lfloor (p \lfloor q) = (a \lfloor p) \lfloor q$.

(f) On interfaces the operator $-$ behaves like relative complementation (or set difference), and hence satisfies the shunting rule $p - q \sqsubseteq r \Leftrightarrow p \sqsubseteq q \sqcup r$.

(g) The restriction $p \lfloor q$ is the greatest lower bound (i.e., the largest common part) of p and q in N. Hence also $p \lfloor q = q \lfloor p$.

Part (a) means that $\lfloor$ really restricts an element a, i.e., the restriction of a to p is contained in a. The same holds for the operator $-$ which is formalised in Part (b). The following two items show that quite natural monotonicity properties hold. Parts (e) and (f) are useful properties that will be used to derive properties presented in the sequel of the paper.

Next we show some important formulas describing the interplay between the both operators.

Lemma 2.4. *We assume again arbitrary code fragments $a, b \in L$ and $p, q \in N$.*

(a) $(a - p) \lfloor p = 0$ *and* $(a \lfloor p) - p = 0$.

(b) $a \sqsubseteq b - p \Leftrightarrow a \sqsubseteq b \wedge a \lfloor p = 0$.

(c) $a \sqsubseteq b \lfloor p \Leftrightarrow a \sqsubseteq b \wedge a - p = 0$.

Part (a) presents annihilation laws. If we subtract from a anything related to p and then restrict exactly to these parts, nothing will remain. In the second equation, a is first restricted. Parts (b) and (c) express that same phenomenon in a way that is more suitable for further proofs.

Finally, as a simple consequence of Lemma 2.4 (c) and the shunting rule (Lemma 2.3)(f) we get

$$p \sqsubseteq q \;\Leftrightarrow\; p \sqsubseteq p \downharpoonright q \;\Leftrightarrow\; p = p \downharpoonright q \,. \tag{11}$$

Now we come to the axiomatic characterisation of the abstract interface function $face$. The key is the observation that $face(X)$ is the least set that leaves X unchanged under the restriction operation $\downharpoonright$:

$$face(X) \subseteq U \Leftrightarrow X = X \downharpoonright U \;,$$

where X is a code fragment and U an abstract interface as in Section **??**. In fact, since $X \downharpoonright U \subseteq U$ holds anyway by definition, this can be relaxed to $X \subseteq X \downharpoonright U \Leftrightarrow face(X) \subseteq U$. Informally, "least" can be interpreted as the fact that the interface has to contain "enough" or "all necessary" information (e.g. all variables of X should be have a counterpart in $face(X)$) but not more (e.g., declarations of variables that do not occur in X should not be in the interface). In that sense the condition $X = X \downharpoonright U$ can also be viewed as a typing assertion saying that X implements the interface U faithfully. This leads to the following algebraic characterisation of the abstract interface.

Definition 2.5 (abstract interface on L)**.** The *abstract interface* f on L is characterised by

$$\mathsf{f}(a) \sqsubseteq p \Leftrightarrow_{df} a \sqsubseteq a \downharpoonright p \;,$$

where $a \in L$, $p \in N$.

The function f behaves in many respects like the abstract codomain operator of [2]. This correspondence allows us to re-use a large body of well-known theory — another advantage of an abstract algebraic view.

As before we present some useful and meaningful consequences of this definition; further properties can again be found in [4]. The first two are immediate from the definition.

Corollary 2.6. *For code fragments $a,b \in L$ and $p,q \in N$ we have $a \sqsubseteq a \downharpoonright \mathsf{f}(a)$ and $\mathsf{f}(a \downharpoonright p) \sqsubseteq p$. In particular $\mathsf{f}(0) = 0$.*

Lemma 2.7. *Assume arbitrary code fragments $a,b \in L$ and $p,q \in N$.*

(a) The abstract interface is additive, i.e, $\mathsf{f}(a) \sqcup \mathsf{f}(b) = \mathsf{f}(a \sqcup b)$. In particular f is isotone.

(b) $\mathsf{f}(p) = p$. In particular, $\mathsf{f}(\mathsf{f}(a)) = \mathsf{f}(a)$.

(c) $a = a \downharpoonright \mathsf{f}(a)$.

(d) $\mathsf{f}(a - p) = \mathsf{f}(a) - p$ and $\mathsf{f}(a \downharpoonright p) = \mathsf{f}(a) \downharpoonright p$.

Part (a) again is a distributivity property. Part (b) means that an abstract interface cannot be abstracted further, since it is abstract already. Part (c) means that the abstract interface of a is no more abstract than necessary: the full a is preserved when restricting it to its abstract interface. Part (d) are import/export laws for bringing extra removals/restrictions in and out of the abstract interface function.

Finally we can characterise the update operator in this abstract setting as follows.

Definition 2.8 (update operator)**.** The *abstract update operator* can now be defined by

$$a \triangleright b =_{df} (b - \mathsf{f}(a)) \sqcup a \,.$$

We call a and b *compatible* if they agree on the common part of their interfaces, i.e., if

$$a \downharpoonright (\mathsf{f}(a) \downharpoonright \mathsf{f}(b)) = b \downharpoonright (\mathsf{f}(a) \downharpoonright \mathsf{f}(b)) \;.$$

As mentioned before, the abstract interface function behaves in many respects like the abstract codomain operator of [2]. Moreover, the update operator is identical to the one of [3] (except that we replace the codomain operator by our interface). Hence we can re-use the theory and get, among others, the following properties of update for free. For the proofs and further laws we refer to [6, 3].

Corollary 2.9. *If a,b,c are arbitrary code fragments, then*

(a) $a = a \triangleright 0 = 0 \triangleright a$.

(b) $(a \triangleright b) \triangleright c = a \triangleright (b \triangleright c)$

(c) The following properties are equivalent:

- *a and b are compatible.*
- $a \lfloor \mathsf{f}(b) = b \lfloor \mathsf{f}(a)$.
- $a \triangleright b = a \sqcup b$,
- $a \triangleright b = b \triangleright a$.

(d) If a and b are compatible then $(a \sqcup b) \triangleright c = a \triangleright (b \triangleright c)$.

(e) If a and b are compatible and $a \triangleright b = b$ then $(a \sqcup b) \triangleright c = a \triangleright c$.

(f) If $\mathsf{f}(a) \lfloor \mathsf{f}(b) = 0$ then $a \triangleright (b \sqcup c) = b \sqcup (a \triangleright c)$.

The first two items show that $(L, \triangleright, 0)$ forms a monoid. In particular there is a neutral element w.r.t. the update operator. In the concrete model this neutral element corresponds to some code fragment without any content. Part (c) means that compatibility is equivalent to the fact that update and join coincide. Part (d) is a sequentialisation property and says that a complex update may also be done by two simpler overwritings if the updates are compatible. Part (e) says that part of an update may be skipped if it would add something that is already present.Part (f) states localisation property, viz. that an update need only be applied to that part of a composition it actually affects.

3 Proofs from [4]

Proof of Lemma 2.3. In this proof we assume that $\lfloor$ and $-$ bind tighter than $\sqcup$.

(a) The claim follows from Axiom (6) by (10).

(b) The first assertion is shown analogously to Part (a). The remaining claim $a \sqsubseteq a - 0$ follows again from splitting (6) and Axiom (5)

$$a = (a \lfloor 0) \sqcup (a - 0) = 0 \sqcup (a - 0) = a - 0 \,.$$

(c) By definition of $\sqsubseteq$, the assumptions are $a \sqcup b = b$ and $p \sqcup q = q$. The claims transform into $(a \lfloor p) \sqcup (b \lfloor p) = b \lfloor p$ and $(a \lfloor p) \sqcup (a \lfloor q) = (a \lfloor q)$, resp., and follow from the distributivity axioms (1) and (2).

$$(a \lfloor p) \sqcup (b \lfloor p) = (a \sqcup b) \lfloor p = b \lfloor p \,,$$
$$(a \lfloor p) \sqcup (a \lfloor q) = a \lfloor (p \sqcup q) = a \lfloor q \,.$$

(d) By definition of $\sqsubseteq$, the assumptions are $a \sqcup b = b$ and $p \sqcup q = q$. The claim for isotony transforms into $(a - p) \sqcup (b - p) = b - p$ which follows from distributivity (3):

$$(a-p) \sqcup (b-p) \;=\; (a \sqcup b) - p \;=\; b - p\ .$$

For the second claim we get by the assumption, Axiom (4) and Part (2)

$$a - q \;=\; a - (p \sqcup q) \;=\; (a-p) - q \sqsubseteq a - p\ .$$

(e) Before showing quasi associativity we derive an auxiliary property. By the splitting axiom (6), exchange law (7), annihilation (8), Axiom (7) and Part (a),

$$q = q{\downharpoonleft}(p-q) \sqcup q-(p-q) \;=\; (q-q){\downharpoonleft}p \sqcup q-(p-q) \;=\; 0{\downharpoonleft}p \sqcup q-(p-q) \;=\; q-(p-q)\ . \quad (*)$$

Now we can prove the associativity property. By the splitting axiom (6) and Equation (4), we first get

$$a{\downharpoonleft}p \;=\; (a{\downharpoonleft}p){\downharpoonleft}q \sqcup (a{\downharpoonleft}p) - q \;=\; a{\downharpoonleft}p){\downharpoonleft}q \sqcup a{\downharpoonleft}(p-q)\ .$$

Similarly, we get $a{\downharpoonleft}p = a{\downharpoonleft}(p{\downharpoonleft}q \sqcup p-q) = a{\downharpoonleft}(p{\downharpoonleft}q) \sqcup a{\downharpoonleft}(p-q)$ and hence

$$(a{\downharpoonleft}p){\downharpoonleft}q \sqcup a{\downharpoonleft}(p-q) \;=\; a{\downharpoonleft}(p{\downharpoonleft}q) \sqcup a{\downharpoonleft}(p-q)\ .$$

We now "subtract" $p-q$ on both sides and get, using distributivity (3),

$$\big(((a{\downharpoonleft}p){\downharpoonleft}q)-(p-q)\big) \sqcup \big((a{\downharpoonleft}(p-q))-(p-q)\big) \;=\; \big((a{\downharpoonleft}(p{\downharpoonleft}q))-(p-q)\big) \sqcup \big((a{\downharpoonleft}(p-q))-(p-q)\big)\ .$$

Now we can apply Lemma 2.4(a) (the proof is given below). This yields

$$((a{\downharpoonleft}p){\downharpoonleft}q)-(p-q) \;=\; (a{\downharpoonleft}(p{\downharpoonleft}q))-(p-q)\ .$$

To show the claim, we simplify both sides of this equation. By Equations (7) and (*) we get

$$((a{\downharpoonleft}p){\downharpoonleft}q)-(p-q) \;=\; (a{\downharpoonleft}p){\downharpoonleft}(q-(p-q)) \;=\; (a{\downharpoonleft}p){\downharpoonleft}q\ .$$

Similarly one can show that $(a{\downharpoonleft}(p{\downharpoonleft}q))-(p-q) = a{\downharpoonleft}(p{\downharpoonleft}q)$ which finally shows the claim.

(f) ($\Rightarrow$) By Axiom (6), the assumption, isotony of $\sqcup$, and Axiom (9) we get

$$p \;=\; p{\downharpoonleft}q \sqcup p-q \sqsubseteq p{\downharpoonleft}q \sqcup r \sqsubseteq q \sqcup r\ .$$

($\Leftarrow$) By isotony and the assumption, Axiom (4), annihilation (8) and Parts (a) and (b), we get

$$p-q \sqsubseteq (q \sqcup r)-q \;=\; q-q \sqcup r-q \;=\; r-q \sqsubseteq r\ .$$

(g) By Axioms (6) and (9) we have $p{\downharpoonleft}q \sqsubseteq p$ and $p{\downharpoonleft}q \sqsubseteq q$, i.e., $p{\downharpoonleft}q$ is a common lower bound of p and q in N. Let now $r \sqsubseteq p,q$ be another common lower bound in N. Then by isotony of ${\downharpoonleft}$ we get $r = r{\downharpoonleft}r \sqsubseteq p{\downharpoonleft}q$, and hence $p{\downharpoonleft}q$ is the greatest lower bound. □

Proof of Lemma 2.4.

(a) By Axioms (7), (8) and (5) we get $(a-p)\downharpoonleft p = a\downharpoonleft(p-p) = a\downharpoonleft 0 = 0$. Similarly, we can show the claim $(a\downharpoonleft p)-p=0$. The remaining claim follows from Axioms (7), (8) and (5)

$$(a\downharpoonleft p)-p = a\downharpoonleft(p-p) = a\downharpoonleft 0 = 0\ .$$

(b) ($\Rightarrow$) The conjunct $a \sqsubseteq b$ follows by isotony of $-$ and transitivity of $\sqsubseteq$, the second conjunct from Part (a), isotony and the assumption

$$a\downharpoonleft p \sqsubseteq (b-p)\downharpoonleft p = 0\ .$$

($\Leftarrow$)We first calculate $a = (a\downharpoonleft p)\sqcup(a-p) = a-p$. This is done using Axiom (6) and the assumption $a\downharpoonleft p \sqsubseteq 0$. The claim then follows by isotony and the other assumption.

(c) Similar to Part (b). □

Proof of Lemma 2.7.

(a) From Lemma 2.4(a) we get $a \sqsubseteq a\downharpoonleft p \Leftrightarrow a-p \sqsubseteq 0$ and hence

$$\mathsf{f}(a) \sqsubseteq p \Leftrightarrow a-p=0\ .$$

By this, Axiom (3) lattice theory and the shunting rule from Lemma 2.3(f), we get, for arbitrary p,

$$\begin{array}{l}
\quad \mathsf{f}(a\sqcup b) \sqsubseteq p \\
\Leftrightarrow (a\sqcup b)-p \sqsubseteq 0 \\
\Leftrightarrow (a-p)\sqcup(b-p) \sqsubseteq 0 \\
\Leftrightarrow a-p \sqsubseteq 0 \ \wedge\ b-p \sqsubseteq 0 \\
\Leftrightarrow \mathsf{f}(a) \sqsubseteq p \ \wedge\ \mathsf{f}(b) \sqsubseteq p \\
\Leftrightarrow \mathsf{f}(a)\sqcup\mathsf{f}(b) \sqsubseteq p\ ,
\end{array}$$

which by the principle of indirect equality, viz.

$$c=d \Leftrightarrow (\forall e : c \sqsubseteq e \Leftrightarrow d \sqsubseteq e)\ ,$$

implies the claim.

(b) Again we use the principle of indirect equality. The claim then follows by definition and Axiom (11):

$$\mathsf{f}(p) \sqsubseteq q \ \Leftrightarrow\ p \sqsubseteq p\downharpoonleft q \ \Leftrightarrow\ p \sqsubseteq q\ .$$

(c) $a \sqsubseteq a\downharpoonleft \mathsf{f}(a)$ follows directly from the definition of abstract interfaces, whereas the other inequality follows from Lemma 2.3(a).

(d) We first use the splitting axiom (6) and get by Part (a) and Corollary 2.6

$$\mathsf{f}(a) = \mathsf{f}((a\downharpoonleft p)\sqcup(a-p)) = \mathsf{f}(a\downharpoonleft p)\sqcup\mathsf{f}(a-p) \sqsubseteq p\sqcup\mathsf{f}(a-p)\ .$$

This implies by shunting (Lemma 2.3(f)) $\mathsf{f}(a)-p \sqsubseteq \mathsf{f}(a-p)$. To show $\mathsf{f}(a-p) \sqsubseteq \mathsf{f}(a)-p$ we use shunting (Lemma 2.3(f)), the definition of f, distributivity (2), isotony, distributivity (3), Lemma 2.4(a), Axiom (7) and the definition again, to calculate, for arbitrary p,

$$\begin{aligned}
&\mathsf{f}(a) - p \sqsubseteq q\\
\Leftrightarrow\ &\mathsf{f}(a) \sqsubseteq q \sqcup p\\
\Leftrightarrow\ &a \sqsubseteq a{\downharpoonright}(q \sqcup p)\\
\Leftrightarrow\ &a \sqsubseteq (a{\downharpoonright}q) \sqcup (a{\downharpoonright}p)\\
\Rightarrow\ &a - p \sqsubseteq ((a{\downharpoonright}q) \sqcup (a{\downharpoonright}p)) - p\\
\Leftrightarrow\ &a - p \sqsubseteq ((a{\downharpoonright}q) - p) \sqcup ((a{\downharpoonright}p) - p)\\
\Leftrightarrow\ &a - p \sqsubseteq (a{\downharpoonright}q) - p\\
\Leftrightarrow\ &a - p \sqsubseteq (a - p){\downharpoonright}q\\
\Leftrightarrow\ &\mathsf{f}(a - p) \sqsubseteq q\ .
\end{aligned}$$

By the principle of indirect inequality

$$c \sqsubseteq d \Leftrightarrow (\forall e : c \sqsubseteq e \Rightarrow d \sqsubseteq e)\ ,$$

this implies $\mathsf{f}(a-p) \sqsubseteq \mathsf{f}(a) - p$.

Next we show $\mathsf{f}(a){\downharpoonright}p \sqsubseteq \mathsf{f}(a{\downharpoonright}p)$ As above we split a by Axiom (6), use Part (a), distributivity (1), the previous result and Lemma 2.3(a)

$$\begin{aligned}
&\mathsf{f}(a){\downharpoonright}p\\
=\ &\mathsf{f}((a{\downharpoonright}p) \sqcup (a-p)){\downharpoonright}p\\
=\ &(\mathsf{f}(a{\downharpoonright}p) \sqcup \mathsf{f}(a-p)){\downharpoonright}p\\
=\ &(\mathsf{f}(a{\downharpoonright}p){\downharpoonright}p) \sqcup (\mathsf{f}(a-p){\downharpoonright}p)\\
=\ &(\mathsf{f}(a{\downharpoonright}p){\downharpoonright}p) \sqcup ((\mathsf{f}(a)-p){\downharpoonright}p)\\
=\ &\mathsf{f}(a{\downharpoonright}p){\downharpoonright}p\\
\sqsubseteq\ &\mathsf{f}(a{\downharpoonright}p)
\end{aligned}$$

Finally we have to prove that $\mathsf{f}(a{\downharpoonright}p) \sqsubseteq \mathsf{f}(a){\downharpoonright}p$. With Lemma 2.4(c), isotony and shunting we get

$$\begin{aligned}
&\mathsf{f}(a{\downharpoonright}p) \sqsubseteq \mathsf{f}(a){\downharpoonright}p\\
\Leftrightarrow\ &\mathsf{f}(a{\downharpoonright}p) \sqsubseteq \mathsf{f}(a)\ \wedge\ \mathsf{f}(a{\downharpoonright}p) - p \sqsubseteq 0\\
\Leftarrow\ &a{\downharpoonright}p \sqsubseteq a\ \wedge\ \mathsf{f}(a{\downharpoonright}p) \sqsubseteq p\ .
\end{aligned}$$

The left part holds by Lemma 2.3(a); the right one by Corollary 2.6. □

References

[1] B. A. Davey and H. A. Priestley. *Introduction to lattices and order,.* Cambridge University Press, 2nd edition, 2002.

[2] J. Desharnais, B. Möller, and G. Struth. Kleene algebra with domain. *ACM Transactions on Computational Logic*, 7(4):798–833, 2006.

[3] T. Ehm. Pointer Kleene algebra. In R. Berghammer, B. Möller, and G. Struth, editors, *RelMiCS*, volume 3051 of *Lecture Notes in Computer Science*, pages 99–111. Springer, 2004.

[4] P. Höfner and B. Möller. An extension of feature algebra. *Science of Computer Programming*, 2010. (submitted).

[5] W. W. McCune. Prover9 and Mace4.
<`http://www.cs.unm.edu/~mccune/prover9`>. (accessed April 27, 2010).

[6] B. Möller. Towards pointer algebra. *Science of Computer Programming*, 21(1):57–90, 1993.

A Prover9 Encodings

```
%--- Operators
  op(500, infix, "+").    %join
  op(490, infix, "I").    %restriction
  op(490, infix, "/").    %complement

  formulas(assumptions).
    %--- Semilattice
      x + y = y + x.
      x + 0 = x.
      x+(y+z) = (x+y)+z.
      x + x = x.
      x <= y <-> x+y=y.

    %--- all about abstract interfaces
      ai(0).
      ai(x) & ai(y) -> ai(x+y).
      ai(x) & ai(y) -> ai(x I y).
      ai(x) & ai(y) -> ai(x/y).

    %--- the original axioms
      all p (ai(p) -> (x+y) I p = (x I p)+(y I p)).                 %(1)
      all p all q (ai(p) & ai(q) -> x I (p+q) = (x I p)+(x I q)).   %(2)
      all p (ai(p) -> (x+y)/p = (x/p)+(y/p)).                       %(3)
      all p all q (ai(p) & ai(q) -> x/(p+q) = (x/p)/q).             %(4)
      x I 0 = 0.                                                    %(5)
      all p (ai(p) -> x = (x I p)+(x/p)).                           %(6)
      all p all q (ai(p) & ai(q) -> x I (q/p) = (x/p) I q).         %(7)
      all p all q (ai(p) & ai(q) -> x I (q/p) = (x I q)/p).         %(7')
      all p (ai(p) -> p/p = 0).                                     %(8)
      all p all q (ai(p) & ai(q) -> p I q <= q ).                   %(9)
  end_of_list.

  formulas(goals).
  end_of_list.
```

B All models up to 8