Algebra of Monotonic Boolean Transformers

Viorel Preoteasa

Åbo Akademi University
Department of Information Technologies
Joukahaisenkatu 3-5 A, 20520 Turku, Finland

Abstract. Algebras of imperative programming languages have been successful in reasoning about programs. In general an algebra of programs is an algebraic structure with programs as elements and with program compositions (sequential composition, choice, skip) as algebra operations. Various versions of these algebras were introduced to model partial correctness, total correctness, refinement, demonic choice, and other aspects. We introduce here an algebra which can be used to model total correctness, refinement, demonic and angelic choice. The basic model of our algebra are monotonic Boolean transformers (monotonic functions from a Boolean algebra to itself).

1 Introduction

Abstract algebra is a useful tool in mathematics. Rather than working with specific models like natural numbers and algebra of truth values, one could reason in a more abstract setting and obtain results which are more general and applicable in different models. Algebras of logics are very important tools in studying various aspects of logical systems. Algebras of programming theories have also a significant contribution to the simplification of reasoning about programs. Programs are elements of an algebra and program compositions and program constants (sequential composition, choice, iteration, skip, fail) are the operations of the algebra. These operations satisfy a number of relations which are used for reasoning about programs. Kleene algebra with tests (KAT) [12] is an extension of Kleene algebra and it is suitable for reasoning about programs in a partial correctness framework. Various versions of Kleene algebras have been introduced, ranging from Kleene algebra with domain [8] and concurrent Kleene algebra [10] to an algebra for separation logic [7].

Refinement Calculus [1,2,6,15] is a calculus based on (monotonic) predicate transformers suitable for program development in a total correctness framework. Within this calculus various aspects of imperative programming languages can be formalized. These include total correctness, partial correctness, demonic choice, and angelic choice. Demonic refinement algebra (DRA) was introduced in [21,22] as a variation of KAT to allow also reasoning about total correctness. The intended model of DRA is the set of conjunctive predicate transformers and this algebra cannot represent angelic choice. General refinement algebra (GRA) was

also introduced in [22], but few results were proved and they were mostly related to iteration. Although the intended model for GRA is the set of monotonic predicate transformers, GRA does not include the angelic choice operator. GRA has been further extended in [20] with enabledness and termination operators, and it was extended for probabilistic programs in [14].

The contribution of this paper is a different extension of GRA with a dual operator [9,4,5,6]. The intended model for our algebra is the set of monotonic Boolean transformers (monotonic functions from a Boolean algebra to itself). In GRA assertions (assumptions) are introduced as disjunctive (conjunctive) elements which have complement. Using the dual operator we characterize these assertion (assumptions) using a conjunction of (in)equations which is simpler than the usual definition from GRA and KAT. We prove that the assertions (assumptions) form a Boolean algebra. Moreover, we also prove that the assertions defined in the algebra are exactly the program assertions in the model of monotonic Boolean transformers. Having the dual operator and the demonic choice operator we automatically obtain also the angelic choice operator. In [20,14] the enabledness and termination operators are introduced using axioms for DRA and GRA respectively. These operators can be defined in our algebra, and their axioms can be proved as theorems.

In DRA [22], a pre-post specification statement is introduced and it is used to prove that a program refines a pre-post specification statement if and only if the program is correct with respect to the pre and post conditions. The proof of this fact requires the assumption that all programs are conjunctive, fact which does not hold for arbitrary monotonic predicate transformers. We have introduced another specification statement, and we proved a similar result in the absence of the conjunctivity property.

The paper is structured as follows. Section 2 introduces the monotonic Boolean transformers that are the model of our algebra. Section 3 introduces the monotonic Boolean transformers algebra and some of its properties. Section 4 introduces the assertions and the assumptions. Some of their properties are also introduced, and proofs that they form Boolean algebras are given. In Section 5, we define the weakest precondition, the guard of a program, Hoare triples [11] for total correctness, data refinement of programs, and we prove some properties of these constructs. The weakest precondition of top satisfies all axioms set for the termination operator in [20,14], and dually the guard of a program satisfies all axioms set for the enabledness operator in [20,14].

All our results were mechanically verified in the Isabelle [16] theorem prover.

2 Monotonic Boolean Transformers

In this section we introduce the concept of monotonic Boolean transformers which is more general than monotonic predicate transformers. For a set of states X, monotonic predicate transformers over X are monotonic functions from $\mathsf{Pred}.X$ to $\mathsf{Pred}.X$ where $\mathsf{Pred}.X = X \to \mathsf{Bool}$ and Bool is the complete Boolean algebra with two elements, true and false. Monotonic predicate transformers are

used for modeling imperative programs. A program is modeled by a predicate transformer S, where if $q \in \mathsf{Pred}.X$ is a predicate (set) of final states, then $S.q$ are the initial states from which the program terminates and if it terminates in a state s, then s is from q.

In this context we only need the assumption that we work with a complete Boolean algebra instead of $\mathsf{Pred}.X$. This generalization is mainly used here because it is sufficient for expressing and proving the properties from this paper. However, we can also apply these results directly to Boolean algebras of the form $X \to Y \to \mathsf{Bool}$, which were used in [3,17,18] for modeling procedures with parameters. Let $\langle B, \wedge, \vee, \leq, \neg, \top, \bot \rangle$ be a complete Boolean algebra. We denote by $\mathsf{Mtran}.B$ the set of all monotonic functions from B to B.

$$\mathsf{Mtran}.B = \{S : B \to B \mid \forall p, q : p \leq q \Rightarrow S.p \leq S.q)\}$$

The elements of $\mathsf{Mtran}.B$ are called monotonic Boolean transformers, or just monotonic transformers, or programs.

We point-wise extend all operations except the negation from B to $\mathsf{Mtran}.B$:

$$\begin{array}{lll} (S \sqcap T).p = S.p \wedge T.p & \mathsf{magic}.p = \top & S \sqsubseteq T = (\forall p : S.p \leq T.p) \\ (S \sqcup T).p = S.p \vee T.p & \mathsf{fail}.p \quad = \bot & \end{array}$$

The extended constants magic and fail are monotonic Boolean transformers and, if S and T are monotonic Boolean transformers, then $S \sqcap T$, and $S \sqcup T$ are monotonic Boolean transformers. We could extend the negation similarly, however the negation applied to a monotonic function is not monotonic.

The program $S \sqcap T$ models the demonic choice between executing S or T. The choice is demonic because the user does not control it. In order for this choice to be correct, both S and T must be correct. The program $S \sqcup T$ models the angelic choice. Here the choice is angelic because the user can decide between executing S or T. This choice is correct if one of the programs S and T is correct. The relation $\sqsubseteq$ is the refinement relation. A program S is refined by a program T if we can replace S by T. The program magic is always correct, but it cannot be implemented. The program fail never terminates. fail is equivalent to $\mathsf{while\ true\ do\ skip}$.

If $S, T \in \mathsf{Mtran}$, $p, q \in \mathsf{Bool}$, then we introduce the transformers $S \circ T$, $\{p\}$, $[p]$, skip, S^{o}, S^{ω}, $||p|| \in \mathsf{Mtran}$, the sequential composition of S and T, the assert statement of p, the assume statement of p, the skip statement, the dual of S, the iteration of S, and the post-condition statement of p, respectively. These are given by the following definitions:

$$\begin{array}{lll} (S \circ T).p = S.(T.p) & & \text{(sequential composition)} \\ \{p\}.q \quad = p \wedge q & & \text{(assert statement)} \\ [p].q \quad = \neg p \vee q & & \text{(assume statement)} \\ \mathsf{skip}.p \quad = p & & \text{(skip statement)} \\ S^{\mathsf{o}}.p \quad = \neg S.(\neg p) & & \text{(dual of a program)} \\ S^{\omega} \quad = \mu\, X : S \circ X \sqcap \mathsf{skip} & & \text{(iteration)} \\ ||p||.q & \begin{cases} \top & \text{if } p \leq q \\ \bot & \text{otherwise} \end{cases} & \text{(postcondition statement)} \end{array}$$

The functional composition of monotonic transformers corresponds to the sequential composition of programs. The assert statement $\{p\}$ executed from a state in which the predicate p is true behaves as skip, otherwise fails. The assume statement $[p]$ executed from a state in which the predicate p is true behaves as skip, otherwise behaves as magic. The statement skip does not change the state of computation. The dual was used in [9,4,5,6] for predicate transformers. The term *conjugate* has also been used to name the dual operator. We will use the dual to define the negation of an assertion in the algebra of monotonic predicate transformers. The iteration is used to define the while program:

$$\mathsf{while}\ b\ \mathsf{do}\ S = ([b] \circ S)^{\omega} \circ [\neg b]$$

The conditional program can be introduced using the assert statement and the angelic choice or using the assume statement and the demonic choice:

$$\mathsf{if}\ b\ \mathsf{then}\ S\ \mathsf{else}\ T = \{b\} \circ S \sqcup \{\neg b\} \circ T = [b] \circ S \sqcap [\neg b] \circ T$$

If $S \in \mathsf{Mtran}$ and $p, q \in B$, then a Hoare triple $p\,\{\!|S|\!\}\,q$ is true if $p \leq S.q$.

The post-condition statement $||p||$ has been used in [17,18] to connect total correctness Hoare triples to refinement statements. The following relation is true

$$p\,\{\!|S|\!\}\,q \ \Leftrightarrow\ \{p\} \circ ||q|| \sqsubseteq S \tag{1}$$

and it is a consequence of the following theorem.

Theorem 1. *If $p \in B$ then*

1. $||p|| \in \mathsf{Mtran}$ ($||p||$ is monotonic)
2. $||p||.p = \top$
3. $\{S.p\} \circ ||p|| \sqsubseteq S$

Definition 1. *A monotonic transformer S is* disjunctive *if for all $p, q \in B$, $S.(p \vee q) = S.p \vee S.q$, which is equivalent to*

$$(\forall U, V \in \mathsf{Mtran} : S \circ (U \sqcup V) = (S \circ U) \sqcup (S \circ V))$$

A monotonic transformer S is conjunctive *if for all $p, q \in B$, $S.(p \wedge q) = S.p \wedge S.q$, which is equivalent to*

$$(\forall U, V \in \mathsf{Mtran} : S \circ (U \sqcap V) = (S \circ U) \sqcap (S \circ V))$$

Next theorem gives a characterization of assertion statements.

Theorem 2. *A monotonic transformer S is an assertion if and only if $S \sqsubseteq \mathsf{skip}$ and S is disjunctive.*

Proof. If S is an assertion it is easy to prove that $S \sqsubseteq \mathsf{skip}$ and S is disjunctive. Conversely, assume that $S \sqsubseteq \mathsf{skip}$ and S is disjunctive. We prove that $S = \{S.\top\}$.

- $\{S.\top\}.q$

$=$ {Boolean algebra property}
$\{S.(q \vee \neg q)\}.q$
$=$ {S is disjunctive}
$\{S.q \vee S.(\neg q)\}.q$
$=$ {Definition of assert and distributivity}
$(S.q \wedge q) \vee (S.(\neg q) \wedge q)$
$=$ {$S \leq \mathsf{skip}$ implies$S.q \leq q$}
$S.q \vee (S.(\neg q) \wedge q)$
$=$ {Prove $S.(\neg q) \wedge q = \bot$}

- $S.(\neg q) \wedge q = \bot$
  $\Leftrightarrow$ {$\bot$ is the least element}
  $S.(\neg q) \wedge q \leq \bot$
  $\Leftarrow$ {$S \leq \mathsf{skip}$ implies$S.(\neg q) \leq \neg q$}
  $\neg q \wedge q \leq \bot$
  $\Leftrightarrow$ {Boolean algebra properties}
  true

$S.q \vee \bot$
$=$ {Boolean algebra properties}
$S.q$ □

Theorem 3. *A monotonic transformer S is an assumption if and only if $S \geq \mathsf{skip}$ and S is conjunctive.*

3 Algebra of monotonic Boolean transformers

We introduce in this section an algebraic structure which has as a model the monotonic Boolean transformers.

Definition 2. *An* algebra of monotonic Boolean transformers *(abbreviated* MBT*) is an algebra $\mathcal{A} = \langle A, \sqcap, \sqcup, \circ, \_^\circ, \_^\omega, 1, \bot, \top \rangle$ where $\sqcap$, $\sqcup$, and $\circ$ are binary operations, $\_^\circ$, $\_^\omega$ are unary operation and 1, $\bot$, and $\top$ are constants, which satisfies the following axioms:*

($A1$) $\langle A, \sqcap, \sqcup, \bot, \top \rangle$ is a bounded distributive lattice
($A2$) $\langle A, \circ, 1 \rangle$ is a monoid

($A3$) $(x \sqcap y) \circ z = (x \circ z) \sqcap (y \circ z)$
($A4$) $x \leq y \Rightarrow z \circ x \leq z \circ y$
($A5$) $\top \circ x = \top$
($A6$) $x \leq y \Leftrightarrow y^\circ \leq x^\circ$
($A7$) $x^{\circ\circ} = x$
($A8$) $(x \circ y)^\circ = x^\circ \circ y^\circ$
($A9$) $(x \circ \top) \sqcap (x^\circ \circ \bot) = \bot$
($A10$) $x^\omega = x \circ x^\omega \sqcap 1$
($A11$) $x \circ z \sqcap y \leq z \Rightarrow x^\omega \circ y \leq z$

The algebra of monotonic Boolean transformers includes all operators and axioms of the general refinement algebra introduced in [22], except the weak iteration operator ($\_^*$) and its axioms. Additionally it includes the angelic choice and the dual operator and their corresponding axioms. We also assume that the lattice of the choice operations is distributive. The iteration operator (ω) is

introduced here only for completion. It will not be used further in this paper. All properties proved in [22] for ω in the general refinement algebra hold also for MBT algebra.

The dual operator behaves like a negation operator: it is anti-monotonic, it is an involution ($x^{\circ\circ} = x$), and the conjunction of $x \circ \top$ and $x^\circ \circ \bot$ is $\bot$. However, the dual operator applied to a monotonic Boolean transformer is also monotonic. This operator will be used to define the negation for assert and assume statements of MBT algebra.

Alternatively we could introduce only $\sqcap$, $\circ$, $\_^\circ$, $\_^\omega$, 1, and $\top$ as primitive operations, and then define $\sqcup$ and $\bot$ in terms of $\sqcap$, $\_^\circ$, and $\top$.

Theorem 4. *If the constants* 1, $\bot$, *and* $\top$ *from* MBT *are interpreted as* skip, fail, *and* magic, *then the monotonic Boolean transformers are a model for the axioms of* MBT.

Proof. All properties $(A1)$ to $(A11)$ from Definition 2 are easy to verify.

In [13], multirelations are used to model angelic and demonic nondeterminism, and they are shown to be equivalent to monotonic predicate transformers. This work would enable showing that the multirelations are also a model for MBT algebra.

Next theorem lists a number of properties that are true in a MBT algebra. The properties are direct consequences of the axioms of MBT algebra.

Theorem 5. *In* MBT *the following properties hold:*

1. $\top^\circ = \bot$ and $\bot^\circ = \top$
2. $1^\circ = 1$
3. $(x \sqcap y)^\circ = x^\circ \sqcup y^\circ$
4. $(x \sqcup y)^\circ = x^\circ \sqcap y^\circ$
5. $x = y \Leftrightarrow x^\circ = y^\circ$
6. $(x \sqcup y) \circ z = (x \circ z) \sqcup (y \circ z)$
7. $x \circ (y \sqcap z) \leq (x \circ y) \sqcap (x \circ z)$
8. $x \circ (y \sqcup z) \geq (x \circ y) \sqcup (x \circ z)$
9. $\bot \circ x = \bot$
10. $x \leq y \Rightarrow x \circ z \leq y \circ z$
11. $x \leq y \wedge u \leq v \Rightarrow x \circ u \leq y \circ v$
12. $1 \leq x \Rightarrow y \leq x \circ y$
13. $1 \leq x \Rightarrow y \leq y \circ x$
14. $x \leq 1 \Rightarrow x \circ y \leq y$
15. $x \leq 1 \Rightarrow y \circ x \leq y$
16. $x \leq x \circ \top$ and $x \circ \bot \leq x$
17. $(x \circ \top) \sqcup (x^\circ \circ \bot) = \top$
18. $(x \circ \bot) \sqcup (x^\circ \circ \top) = \top$

Definition 3. *An element* x is conjunctive *if it satisfies*

$$(\forall y, z : x \circ (y \sqcap z) = (x \circ y) \sqcap (x \circ z))$$

and dually x *is* disjunctive *if it satisfies*

$$(\forall y, z : x \circ (y \sqcup z) = (x \circ y) \sqcup (x \circ z))$$

The set of conjunctive and disjunctive elements are denoted by Conj *and* Disj, *respectively.*

As pointed out in Section 2, these definitions are equivalent to the definitions of conjunctive and disjunctive functions in the model of monotonic Boolean transformers.

Lemma 1. *For $x \in \mathsf{MBT}$ the following properties hold*

1. $x \in \mathsf{Conj} \Rightarrow x^\circ \in \mathsf{Disj}$
2. $x \in \mathsf{Disj} \Rightarrow x^\circ \in \mathsf{Conj}$

4 Assertions and assumptions

This section introduces the set of assertions and assumptions of a MBT algebra. In a Kleene algebra with tests [12], the tests (which are the equivalent to assumptions) are postulated. The tests are elements of a subset of a Kleene algebra and they form a Boolean algebra. In a demonic refinement algebra [22], guards (which are equivalent to assumptions) are the elements that have a complement with respect to $\sqcap$, $\circ$,1, and $\top$. Because our algebra contains the dual operator we are able to introduce the assertions using a conjunction of an inequality and an equality which is logically simpler than the definition from [22]. We prove that the assertions and also the assumptions are Boolean algebras, and moreover, we prove that the assertions and the assumptions from our algebra correspond exactly to the assertions and the assumptions from the monotonic Boolean transformers model.

Definition 4. *In a MBT algebra the set of assertions is defined by*

$$\mathsf{Assertion} = \{p : p \leq 1 \wedge p = (p \circ \top) \sqcap p^\circ\}$$

Lemma 2. *Let $p \in \mathsf{Assertion}$ then*

1. $p^\circ = (p^\circ \circ \bot) \sqcup p$
2. $p = (p \circ \top) \sqcap 1$ and $p^\circ = (p^\circ \circ \bot) \sqcup 1$
3. $p, p^\circ \in \mathsf{Conj}$ and $p, p^\circ \in \mathsf{Disj}$

Proof. We prove only the last property. First we prove $p \in \mathsf{Conj}$, i.e. for all $x, y \in \mathsf{MBT}$, $p \circ (x \sqcap y) = (p \circ x) \sqcap (p \circ y)$:

$$\begin{array}{ll}
\bullet & p \circ (x \sqcap y) \\
= & \{\text{property 2. of this theorem}\} \\
& ((p \circ \top) \sqcap 1) \circ (x \sqcap y) \\
= & \{\text{axioms of MBT}\} \\
& (p \circ \top \circ (x \sqcap y)) \sqcap x \sqcap y \\
= & \{\text{axioms of MBT}\} \\
& (p \circ \top) \sqcap x \sqcap y \\
= & \{\text{lattice properties}\} \\
& ((p \circ \top) \sqcap x) \sqcap ((p \circ \top) \sqcap y) \\
= & \{\text{axioms of MBT}\} \\
& (((p \circ \top) \sqcap 1) \circ x) \sqcup (((p \circ \top) \sqcap 1) \circ y) \\
= & \{\text{property 2. of this theorem}\} \\
& (p \circ x) \sqcap (p \circ y)
\end{array}$$

The property $p \in \mathsf{Disj}$ can be proved similarly, but we also need to use the distributivity of $\sqcap$ over $\sqcup$.

Finally $p^\circ \in \mathsf{Conj}$ and $p^\circ \in \mathsf{Disj}$ follow using Lemma 1. □

The definition of assertions corresponds to assertions in the model of Boolean transformers.

Theorem 6. *In* Mtran *the set* Assertion *is the set of all assertions,* $\{p\}$*, for* $p \in \mathsf{Bool}$.

Proof. We prove that $\mathsf{Assertion} = \{\{p\} \mid p \in \mathsf{Bool}\}$ in Mtran. First if $p \in \mathsf{Bool}$, it is easy to show that $\{p\} \in \mathsf{Assertion}$. Conversely if $x \in \mathsf{Assertion}$, then by Lemma 5 $x \in \mathsf{Disj}$, and using Theorem 2, it follows that $x \in \{\{p\} \mid p \in \mathsf{Bool}\}$. □

Lemma 3. *If* $p, q \in \mathsf{Assertion}$*, then* $p \circ q = p \sqcap q$.

Proof. The inequality $p \circ q \leq p \sqcap q$ follows directly from the axioms of MBT.

The second inequality follows from:

$$
\begin{array}{ll}
\bullet & p \sqcap q \\
= & \{\mathsf{Assertion}\ \text{definition}\} \\
& p \circ \top \sqcap p^\circ \sqcap q \circ \top \sqcap q^\circ \\
\leq & \{\text{sub-derivation}\} \\
& \quad \bullet \quad p^\circ \leq p^\circ \circ q^\circ \wedge q^\circ \leq p^\circ \circ q^\circ \\
& \quad = \quad \{\text{Theorem 5}\} \\
& \quad \quad \mathsf{true} \\
& \quad \bullet \quad q \leq p^\circ \circ q \circ \top \\
& \quad \Leftarrow \quad \{\text{transitivity of} \leq\} \\
& \quad \quad q \leq p^\circ \circ q \wedge p^\circ \circ q \leq p^\circ \circ q \circ \top \\
& \quad = \quad \{\text{Theorem 5}\} \\
& \quad \quad \mathsf{true} \\
& p \circ \top \sqcap p^\circ \circ q \circ \top \sqcap p^\circ \circ q^\circ \\
= & \{p^\circ \in \mathsf{Conj}\} \\
& p \circ \top \sqcap p^\circ \circ (q \circ \top \sqcap q^\circ) \\
= & \{q \in \mathsf{Assertion}\} \\
& p \circ \top \sqcap p^\circ \circ q \\
= & \{\mathsf{MBT}\ \text{axioms}\} \\
& (p \circ \top \sqcap p^\circ) \circ q \\
= & \{p \in \mathsf{Assertion}\} \\
& p^\circ \circ q^\circ
\end{array}
$$

This concludes the theorem. □

Definition 5. *For an assertion* $p \in \mathsf{Assertion}$ *the* negation *of* p*, denoted* $\neg p$ *is defined by*

$$\neg p = (p^\circ \circ \bot) \sqcap 1$$

Theorem 7. *The assertions are closed under* $\sqcap$, $\sqcup$, $\neg$, 1, *and* $\bot$.

Proof. We prove only that $p \sqcap q \in \mathsf{Assertion}$. It is true that $p \sqcap q \leq 1$. We prove also that $p \sqcap q = (p \sqcap q) \circ \top \sqcap (p \sqcap q)^\circ$:

$$
\begin{array}{ll}
\bullet & p \sqcap q \\
= & \{\text{Lemma 3}\} \\
& p \circ q \\
= & \{p, q \in \mathsf{Assertion}\} \\
& (p \circ \top \sqcap p^{\circ}) \circ (q \circ \top \sqcap q^{\circ}) \\
= & \{\text{axioms of } \mathsf{MBT}\} \\
& p \circ \top \sqcap p^{\circ} \circ (q \circ \top \sqcap q^{\circ}) \\
= & \{p^{\circ} \in \mathsf{Conj} \text{ by Lemma 2}\} \\
& p \circ \top \sqcap p^{\circ} \circ q \circ \top \sqcap p^{\circ} \circ q^{\circ} \\
= & \{\mathsf{MBT} \text{ axioms}\} \\
& (p \circ \top \sqcap p^{\circ}) \circ q \circ \top \sqcap p^{\circ} \circ q^{\circ} \\
= & \{p \in \mathsf{Assertion}\} \\
& p \circ q \circ \top \sqcap p^{\circ} \circ q^{\circ} \\
= & \{\mathsf{MBT} \text{ axioms}\} \\
& p \circ q \circ \top \sqcap (p \circ q)^{\circ} \\
= & \{\text{Lemma 3}\} \\
& (p \sqcap q) \circ \top \sqcap (p \sqcap q)^{\circ}. \qquad \square
\end{array}
$$

Theorem 8. *The structure* $(\mathsf{Assertion}, \sqcap, \sqcup, \neg, \bot, 1)$ *is a Boolean algebra.*

Proof. The structure $(\mathsf{Assertion}, \sqcap, \sqcup, \bot, 1)$ is a bounded distributive lattice by Theorem 7 and by the fact that MBT is a distributive lattice. We need to show also that $\neg$ satisfies the negation axioms: $p \sqcap \neg p = \bot$ and $p \sqcup \neg p = 1$. We only show here $p \sqcup \neg p = 1$.

$$
\begin{array}{ll}
\bullet & p \sqcup \neg p \\
= & \{\text{Definition of } \neg\} \\
& p \sqcup (p^{\circ} \circ \bot \sqcap 1) \\
= & \{\text{lattice distributivity}\} \\
& (p \sqcup p^{\circ} \circ \bot) \sqcap (p \sqcup 1) \\
= & \{p \in \mathsf{Assertion}\} \\
& (p \sqcup p^{\circ} \circ \bot) \sqcap 1 \\
= & \{\text{Theorem 5 and } \mathsf{MBT} \text{ axioms}\} \\
& (p^{\circ} \sqcap p \circ \top)^{\circ} \sqcap 1 \\
= & \{p \in \mathsf{Assertion}\} \\
& p^{\circ} \sqcap 1 \\
= & \{p \leq 1 \Leftrightarrow 1 \leq p^{\circ} \text{ by } \mathsf{MBT} \text{ axioms}\} \\
& 1 \qquad \square
\end{array}
$$

Next lemma introduces some additional properties for assertions.

Lemma 4. *If* $p \in \mathsf{Assertion}$ *and* $x, y \in \mathsf{MBT}$*, then*

1. $p \circ p = p$ and $p^{\circ} \circ p^{\circ} = p^{\circ}$
2. $p \circ p^{\circ} = p$ and $p^{\circ} \circ p = p^{\circ}$
3. $p^{\circ} \circ x \sqcup (\neg p) \circ \top = p^{\circ} \circ x$
4. $p \circ x \sqcup (\neg p) \circ y = p^{\circ} \circ x \sqcap (\neg p)^{o} \circ y$

Proof. We prove only the last property:

$$
\begin{array}{ll}
\bullet & p \circ x \sqcup (\neg p) \circ y \\
= & \{p, \neg p \in \mathsf{Assertion}\} \\
& (p \circ \top \sqcap p^{\circ}) \circ x \sqcup ((\neg p) \circ \top \sqcap (\neg p)^{\circ}) \circ y \\
= & \{\mathsf{MBT}\ \text{axioms}\} \\
& (p \circ \top \sqcap p^{\circ} \circ x) \sqcup ((\neg p) \circ \top \sqcap (\neg p)^{\circ} \circ y) \\
= & \{\text{lattice distributivity}\} \\
& ((p \circ \top \sqcap p^{\circ} \circ x) \sqcup (\neg p) \circ \top) \sqcap ((p \circ \top \sqcap p^{\circ} \circ x) \sqcup (\neg p)^{\circ} \circ y) \\
= & \{\text{lattice distributivity}\} \\
& (p \circ \top \sqcup (\neg p) \circ \top) \sqcap (p^{\circ} \circ x \sqcup (\neg p) \circ \top) \sqcap (p \circ \top \sqcup (\neg p)^{\circ} \circ y) \sqcap (p^{\circ} \circ x \sqcup (\neg p)^{\circ} \circ y) \\
= & \{\mathsf{MBT}\ \text{axioms}\} \\
& (p \sqcup (\neg p)) \circ \top \sqcap (p^{\circ} \circ x \sqcup (\neg p) \circ \top) \sqcap (p \circ \top \sqcup (\neg p)^{\circ} \circ y) \sqcap (p^{\circ} \circ x \sqcup (\neg p)^{\circ} \circ y) \\
= & \{\mathsf{Assertion}\ \text{is a Boolean algebra}\} \\
& (p^{\circ} \circ x \sqcup (\neg p) \circ \top) \sqcap (p \circ \top \sqcup (\neg p)^{\circ} \circ y) \sqcap (p^{\circ} \circ x \sqcup (\neg p)^{\circ} \circ y \\
= & \{\text{Property 3: } p^{\circ} \circ x \sqcup (\neg p) \circ \top = p^{\circ} \circ x\} \\
& p^{\circ} \circ x \sqcap (\neg p)^{\circ} \circ y \sqcap (p^{\circ} \circ x \sqcup (\neg p)^{\circ} \circ y) \\
= & \{\text{lattice properties}\} \\
& p^{\circ} \circ x \sqcap (\neg p)^{\circ} \circ y
\end{array}
$$

□

The property 3. from Lemma 4 shows that the two ways of defining the conditional program are also equivalent in MBT. In MBT the conditional program is defined by

$$\mathsf{if}\ b\ \mathsf{then}\ x\ \mathsf{else}\ y = b \circ x \sqcup \neg b \circ y = b^{\circ} \circ x \sqcap (\neg b)^{\circ} \circ y$$

The assumptions of MBT are defined similarly to assertions, but using the duals of the properties for assertions.

Definition 6. *The assumptions of* MBT*, denoted by* Assumption $\subseteq$ MBT*, are defined by*

$$\mathsf{Assumption} = \{g : 1 \leq g \wedge g = (g \circ \bot) \sqcup g^{\circ}\}$$

Lemma 5. *Let* $g \in$ Assumption *then*

1. $g \in \mathsf{Assumption} \Leftrightarrow g^{\circ} \in \mathsf{Assertion}$
2. $g^{\circ} = (g^{\circ} \circ \top) \sqcap g$
3. $g = (g \circ \bot) \sqcup 1$ and $g^{\circ} = (g^{\circ} \circ \top) \sqcap 1$
4. $g, g^{\circ} \in \mathsf{Conj}$ and $g, g^{\circ} \in \mathsf{Disj}$

Proof. These properties can be proved similarly to those for assertion, but using the dual properties. □

Theorem 9. *In* Mtran *the* Assumption *is the set of all assumptions,* $[p]$*, for* $p \in B$*.*

Proof. This fact can be proved similarly to Theorem 6, using Theorem 3. This theorem can also be proved using Lemma 5.1, and the fact that in Mtran $\{p\}^{\circ} = [p]$. □

The negation of an assumption can be defined using the negation of an assertion.

Definition 7. *The negation of an assumption* $g \in \mathsf{Assumption}$*, denoted* $\neg g \in \mathsf{Assumption}$*, is given by*

$$\neg q = (\neg g^{\circ})^{\circ}$$

Theorem 10. *The assumptions are closed to the operations* $\sqcap$*,* $\sqcup$*,* $\neg$*,* 1*, and* $\top$*, and the structure* $(\mathsf{Assumption}, \sqcap, \sqcup, \neg, 1, \top)$ *is a Boolean algebra.*

5 Weakest precondition, guards, Hoare triples, and data refinement.

This sections introduces the weakest precondition for elements of a MBT algebra, and using it introduces valid Hoare triples. Various results connecting valid Hoare triples, refinement, and data refinement are also proved.

Definition 8. *The* weakest precondition *of a program* x *and* $\top$*, denoted* $\mathsf{wpt}.x \in \mathsf{MBT}$*, is given by*

$$\mathsf{wpt}.x = (x \circ \top) \sqcap 1.$$

This definition is justified by the fact that in the monotonic Boolean transformer model $\mathsf{wpt}.S$ is equal to $\{S.\top\}$ and $\mathsf{wpt}.(S \circ \{p\})$ is equal to $\{S.p\}$. The operator wpt satisfies all axioms set for the termination operator in [20,14]. These axioms are listed among the conclusions of the next theorem.

Theorem 11. *The following properties are true for* wpt.

1. $\mathsf{wpt}.x \in \mathsf{Assertion}$
2. $(\mathsf{wpt}.x) \circ x = x$
3. $p \in \mathsf{Assertion} \Rightarrow \mathsf{wpt}.p = p$
4. $p \in \mathsf{Assertion} \wedge p \circ x = x \Rightarrow \mathsf{wpt}.x \leq p$
5. $p \in \mathsf{Assertion} \Rightarrow \mathsf{wpt}.(p^{\circ}) = 1$
6. $p, q \in \mathsf{Assertion} \Rightarrow \mathsf{wpt}.(p^{\circ} \circ q) = \neg p \sqcup q$
7. $x \leq y \Rightarrow \mathsf{wpt}.x \leq \mathsf{wpt}.y$
8. $\mathsf{wpt}.(x \circ y) = \mathsf{wpt}.(x \circ \mathsf{wpt}.y)$
9. $p \in \mathsf{Assertion} \wedge x \in \mathsf{Conj} \Rightarrow x \circ p = \mathsf{wpt}.(x \circ p) \circ x$ (moving assertions)
10. $(\mathsf{wpt}.x) \circ \top = x \circ \top$

Proof. We only show here the proof of property 6.

$$\begin{array}{ll}
\bullet & \mathsf{wpt}.(p^{\circ} \circ q) \\
= & \{\text{Definition}\} \\
& (p^{\circ} \circ q \circ \top) \sqcap 1 \\
= & \{\text{Lemma 2}\} \\
& (((p^{\circ} \circ \bot) \sqcup 1) \circ q) \sqcap 1 \\
= & \{\mathsf{MBT}\text{ axioms}\} \\
& ((p^{\circ} \circ \bot) \sqcup q) \sqcap 1 \\
= & \{\text{lattice distributivity}\} \\
& ((p^{\circ} \circ \bot) \sqcap 1) \sqcup (q \sqcap 1)
\end{array}$$

$=$ $\{q \in \mathsf{Assertion}\}$
$((p^{\circ} \circ \bot) \sqcap 1) \sqcup q$
$=$ {definition of $\neg$}
$\neg p \sqcup q$ □

In Mtran the guard of a program is defined as the set of all states from which the program is guaranteed to terminate. Formally in Mtran the guard of a program S is the predicate $\neg S.\bot$. In MBT we can also define the guard of a program as an assumption.

Definition 9. *The guard of an element $x \in \mathsf{MBT}$, denoted $\mathsf{grd}.x$, is given by*

$$\mathsf{grd}.x = x \circ \bot \sqcup 1.$$

In Mtran the guard of a program S corresponds to $[\neg S.\bot]$: $\mathsf{grd}.S = [\neg S.\bot]$

The operator grd satisfies all axioms set for the termination operator in [20,14]. These axioms are listed among the conclusions of the next theorem.

Theorem 12. *If $x \in \mathsf{MBT}$, and $p \in \mathsf{Assertion}$, then*

1. $\mathsf{grd}.x \in \mathsf{Assumption}$
2. $\mathsf{grd}.x \circ x = x$
3. $\mathsf{grd}.x = (\neg \mathsf{wpt}.(x \circ \bot))^{\circ}$
4. $g \in \mathsf{Assumption} \Rightarrow g \leq \mathsf{grd}.(g \circ x)$
5. $\mathsf{grd}.(x \circ y) = \mathsf{grd}.(x \circ \mathsf{grd}.y)$
6. $(\mathsf{grd}.x) \circ \bot = x \circ \bot$

Proof. We prove only the property 3 here:

• $(\neg \mathsf{wpt}.(x \circ \bot))^{\circ}$
$=$ {definition of $\neg$}
$((\mathsf{wpt}.(x \circ \bot))^{\circ} \circ \bot \sqcap 1)^{\circ}$
$=$ {Theorem 5 and MBT axioms}
$(\mathsf{wpt}.(x \circ \bot)) \circ \top \sqcup 1$
$=$ {definition of wpt}
$(x \circ \bot \circ \top \sqcap 1) \circ \top \sqcup 1$
$=$ {Theorem 5 and MBT axioms}
$x \circ \bot \sqcup 1$
$=$ {definition of grd}
$\mathsf{grd}.x$. □

Definition 10. *For $p, q, x \in \mathsf{MBT}$, the* Hoare total correctness triple $p \,\{\!|x|\!\}\, q \in \mathsf{Bool}$ *is defined by*

$$(p \,\{\!|x|\!\}\, q) := p \leq \mathsf{wpt}.(x \circ q).$$

This definition also corresponds to the classical definition of Hoare total correctness triples in the monotonic Boolean transformers lattice. If $p, q \in \mathsf{Bool}$ and $S \in \mathsf{Mtran}$, then $\{p\}\,\{\!|S|\!\}\,\{q\}$ is equivalent to $p \leq S.q$.

In [22] the total correctness triple of a program x with respect to a precondition p and a post-condition q is defined by $p^{\circ} \circ x \circ (-q)^{\circ} = \top$. Next theorems shows that this definition is equivalent to our definition.

Theorem 13. *If* $p \in \mathsf{Assertion}$ *then*

$$p \,\{\!|x|\!\}\, q \;\Leftrightarrow\; p^{\circ} \circ x \circ (\neg q)^{\circ} = \top$$

Proof. First assume $p \,\{\!|x|\!\}\, q$, which implies $p \circ \top \leq x \circ q \circ \top$. Show $p^{\circ} \circ x \circ (\neg q)^{\circ} = \top$.

$$\begin{array}{ll}
\bullet & \top \\
= & \{\text{Theorem 5}\} \\
 & (x \circ q)^{\circ} \circ \bot \sqcup x \circ q \circ \top \\
\leq & \{\text{the assumption implies } p \circ \top \leq x \circ q \circ \top\} \\
 & p^{\circ} \circ \bot \sqcup x \circ q \circ \top \\
= & \{\text{Theorem 5}\} \\
 & (p^{\circ} \circ \bot \sqcup 1) \circ x \circ q \circ \top \\
= & \{\text{Lemma 2}\} \\
 & p^{\circ} \circ x \circ q \circ \top \\
\leq & \{\mathsf{MBT}\text{ axioms}\} \\
 & p^{\circ} \circ x \circ (q \circ \top \sqcup 1) \\
= & \{\text{Theorem 5}\} \\
 & p^{\circ} \circ x \circ (q^{\circ} \circ \bot \sqcap 1)^{\circ} \\
= & \{\text{definition of } \neg\} \\
 & p^{\circ} \circ x \circ (\neg q)^{\circ}
\end{array}$$

For the second implication assume $p^{\circ} \circ x \circ (\neg q)^{\circ} = \top$, which is equivalent to $p^{\circ} \circ x \circ (q \circ \top \sqcup 1) = \top$. To show $p \,\{\!|x|\!\}\, q$ it is enough to show $p \leq x \circ q \circ \top$, which follows from $p \leq p \circ \top = p \circ x \circ q \circ \top \leq x \circ q \circ \top$.

$$\begin{array}{ll}
\bullet & p \circ \top \\
= & \{\mathsf{MBT}\text{ axioms}\} \\
 & p \circ \top \circ \bot \\
= & \{\text{assumption}\} \\
 & p \circ p^{\circ} \circ x \circ (q \circ \top \sqcup 1) \circ \bot \\
= & \{\text{Lemma 4}\} \\
 & p \circ x \circ (q \circ \top \sqcup 1) \circ \bot \\
= & \{\text{Theorem 5}\} \\
 & p \circ x \circ q \circ \top
\end{array}$$

□

Definition 11. *For* $x, y, u, v \in \mathsf{MBT}$*, the program* x *is* data refined *by the program* y *via the programs* u *and* v*, denoted* $x \sqsubseteq_{u,v} y$*, if*

$$u \circ x \leq y \circ v.$$

This definition for data refinement was used in [19] for constructing invariant based programs using data refinement. Next theorem allows to conclude a correctness statement for a program y which data refines a program x, knowing that x is correct.

Theorem 14. *If* $p, x, y, q, u, v \in \mathsf{MBT}$*, then*

1. $p \,\{\!|x|\!\}\, q \wedge x \sqsubseteq_{u,v} y \Rightarrow \mathsf{wpt}.(u \circ p)\,\{\!|y|\!\}\,\mathsf{wpt}.(v \circ q)$
2. $p \in \mathsf{Assertion} \wedge p \,\{\!|x|\!\}\, q \wedge p \circ x \sqsubseteq_{u,v} y \Rightarrow \mathsf{wpt}.(u \circ p)\,\{\!|y|\!\}\,\mathsf{wpt}.(w \circ q)$

Proof. We prove only the second property. The first one has a similar proof. Assume $p \,\{\!|x|\!\}\, q$ ($\Leftrightarrow p \leq \mathsf{wpt}.(x \circ q)$) and $p \circ x \sqsubseteq_{u,v} y$ ($\Leftrightarrow u \circ p \circ x \leq y \circ v$).

$$
\begin{array}{ll}
\bullet & \mathsf{wpt}.(u \circ p)\,\{\!|y|\!\}\,\mathsf{wpt}.(v \circ q) \\
= & \{\text{definition of Hoare triple}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(y \circ \mathsf{wpt}.(v \circ q)) \\
= & \{\text{Theorem 11}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(y \circ v \circ q) \\
\Leftarrow & \{\text{assumption and wpt monotonic}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(u \circ p \circ x \circ q) \\
= & \{\text{Theorem 11}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(u \circ p \circ \mathsf{wpt}.(x \circ q)) \\
\Leftarrow & \{\text{assumption and wpt monotonic}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(u \circ p \circ p) \\
= & \{\text{Lemma 4}\} \\
& \mathsf{wpt}.(u \circ p) \leq \mathsf{wpt}.(u \circ p) \\
= & \{\leq \text{ is reflexive}\} \\
& \mathsf{true}
\end{array}
$$

□

The second property of Theorem 14 is preferable to the first one because the data refinement $p \circ x \sqsubseteq_{u,v} y$ is easier to prove compared to $x \sqsubseteq_{u,v} y$. In $p \circ x \sqsubseteq_{u,v} y$ the properties from p can be used as assumption in the proof.

In [22], von Wright uses a statement called havoc to introduce a pre-post-condition specification statement. von Wright proves that the specification statement is refined by another program x if and only if x is totally correct with respect to the pre and post conditions. However the proof from [22] uses the property that all programs are conjunctive, which does not hold in our setting. We introduce another concept that can be used to define the specification statement and we can prove the equivalence between the refinement of the specification statement into x and the correctness statement of x. As in case of [22], this concept cannot be defined and we use two axioms for introducing it. We assume that we have a function $|\_| : \mathsf{Assertion} \to \mathsf{MBT}$ that satisfies the additional axioms:

$$(P1) \quad |p| \circ p \circ \top = \top \qquad (P2) \quad x \circ p \circ \top \sqcap |p| \leq x$$

In the model of monotonic Boolean transformers, if we define $|\{p\}| = ||p||$, then the axioms $(P1)$ and $(P2)$ are satisfied.

Theorem 15. *If* $p, q \in \mathsf{Assertion}$ *and* $x \in \mathsf{MBT}$*, then* $p \,\{\!|x|\!\}\, q \Leftrightarrow p \circ |q| \leq x$.

Proof. Assume $p \,\{\!|x|\!\}\, q$ which is equivalent to $p \leq x \circ q \circ \top$.

$$
\begin{array}{ll}
\bullet & p \circ |q| \\
= & \{\text{Lemma 2}\} \\
& (p \circ \top \sqcap 1) \circ |q| \\
= & \{\mathsf{MBT} \text{ axioms}\}
\end{array}
$$

$$
\begin{array}{ll}
 & p \circ \top \sqcap |q| \\
\leq & \{\text{assumption}\} \\
 & x \circ q \circ \top \sqcap |q| \\
\leq & \{\text{axiom } (P2)\} \\
 & x
\end{array}
$$

Conversely assume $p \circ |q| \leq x$ and show $p \leq x \circ q \circ \top$ which is equivalent to $p \,\{\!|x|\!\}\, q$ when $p \in \mathsf{Assertion}$

$$
\begin{array}{ll}
\bullet & p \\
= & \{\text{Lemma 2}\} \\
 & (p \circ \top \sqcap 1) \\
= & \{\text{axiom } (P1)\} \\
 & p \circ |q| \circ q \circ \top \sqcap 1 \\
\leq & \{\text{lattice properties}\} \\
 & p \circ |q| \circ q \circ \top \\
\leq & \{\text{assumption}\} \\
 & x \circ q \circ \top
\end{array}
$$

□

6 Conclusions

We have introduced a new algebra for reasoning about imperative programming languages which supports total correctness, refinement, data refinement, demonic choice, and angelic choice. Compared to earlier versions of program algebras, this approach uses the dual of a program as a primitive operation, and the assertion statements are defined using weaker properties than how they were defined in previous work.

We proved a number of results about assertions and assumptions. We have also proved two main theorems. One theorem states a result which can be used to prove the correctness of a concrete program y, by proving that y data refines a program x and x is correct. The other theorem shows the equivalence between the refinement of a specification statement and a Hoare total correctness triple.

All results presented in this paper were mechanically verified in the Isabelle theorem prover.

References

1. R.-J. Back. *On the correctness of refinement in program development.* PhD thesis, Department of Computer Science, University of Helsinki, 1978.
2. R.-J. Back. *Correctness preserving program refinements: proof theory and applications*, volume 131 of *Mathematical Centre Tracts.* Mathematisch Centrum, Amsterdam, 1980.
3. R.-J. Back and V. Preoteasa. An algebraic treatment of procedure refinement to support mechanical verification. *Formal Aspects of Computing*, 17:69 – 90, May 2005.

4. R.-J. Back and J. von Wright. A lattice-theoretical basis for a specification language. In *Proceedings of the International Conference on Mathematics of Program Construction, 375th Anniversary of the Groningen University*, pages 139–156, London, UK, 1989. Springer-Verlag.
5. R.-J. Back and J. von Wright. Duality in specification languages: a lattice-theoretical approach. *Acta Inf.*, 27:583–625, July 1990.
6. R.-J. Back and J. von Wright. *Refinement Calculus. A systematic Introduction.* Springer, 1998.
7. H.-H. Dang, P. Höfner, and B. Möller. Algebraic separation logic. *Journal of Logic and Algebraic Programming*, 80(6):221 – 247, 2011. Relations and Kleene Algebras in Computer Science.
8. J. Desharnais, B. Möller, and G. Struth. Kleene algebra with domain. *ACM Trans. Comput. Logic*, 7:798–833, October 2006.
9. P. Guerreiro. Another characterization of weakest preconditions. In M. Dezani-Ciancaglini and U. Montanari, editors, *International Symposium on Programming*, volume 137 of *Lecture Notes in Computer Science*, pages 164–177. Springer Berlin / Heidelberg, 1982. 10.1007/3-540-11494-7_12.
10. C. A. Hoare, B. Möller, G. Struth, and I. Wehrman. Concurrent kleene algebra. In *Proceedings of the 20th International Conference on Concurrency Theory*, CONCUR 2009, pages 399–414, Berlin, Heidelberg, 2009. Springer-Verlag.
11. C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.
12. D. Kozen. Kleene algebra with tests. *ACM Trans. Program. Lang. Syst.*, 19:427–443, May 1997.
13. C. E. Martin, S. A. Curtis, and I. Rewitzky. Modelling angelic and demonic nondeterminism with multirelations. *Science of Computer Programming*, 65(2):140 – 158, 2007. Special Issue dedicated to selected papers from the conference of program construction 2004 (MPC 2004).
14. L. Meinicke and K. Solin. Refinement algebra for probabilistic programs. *Formal Aspects of Computing*, 22:3–31, 2010. 10.1007/s00165-009-0111-1.
15. C. Morgan. *Programming from specifications.* Prentice-Hall, Inc., 1990.
16. T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.
17. V. Preoteasa. *Program Variables – The Core of Mechanical Reasoning about Imperative Programs.* PhD thesis, Turku Centre for Computer Science, Nov 2006.
18. V. Preoteasa. Frame rule for mutually recursive procedures manipulating pointers. *Theoretical Computer Science*, 410(42):4216 – 4233, 2009.
19. V. Preoteasa and R.-J. Back. Data refinement of invariant based programs. *Electronic Notes in Theoretical Computer Science*, 259:143 – 163, 2009. Proceedings of the 14th BCS-FACS Refinement Workshop (REFINE 2009).
20. K. Solin and J. von Wright. Enabledness and termination in refinement algebra. *Sci. Comput. Program.*, 74:654–668, June 2009.
21. J. von Wright. From kleene algebra to refinement algebra. In *Proceedings of the 6th International Conference on Mathematics of Program Construction*, MPC '02, pages 233–262, London, UK, UK, 2002. Springer-Verlag.
22. J. von Wright. Towards a refinement algebra. *Sci. Comput. Program.*, 51:23–45, May 2004.