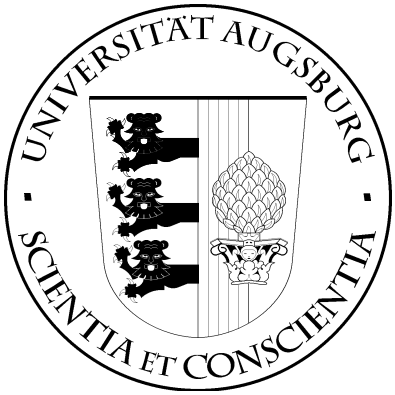
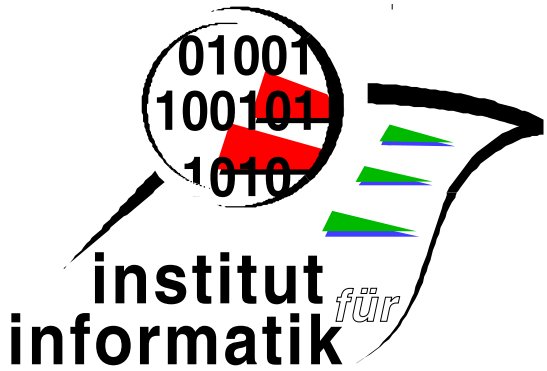
Universität Augsburg



Kleene Under a Demonic Star

Jules Desharnais Bernhard Möller Fairouz Tchier

Report 2000-3 Februar 2000



Institut für Informatik
D-86135 Augsburg

Kleene Under a Demonic Star

Jules Desharnais[1], Bernhard Möller[2], and Fairouz Tchier[3]

[1] Département d'informatique, Université Laval, Québec QC G1K 7P4 Canada
Jules.Desharnais@ift.ulaval.ca
[2] Institut für Informatik, Universität Augsburg, D-86135 Augsburg, Germany
Bernhard.Moeller@informatik.uni-augsburg.de
[3] Mathematics Department, King Saud University, P.O.Box 22452, Riyadh 11495, Saudi Arabia, f66m002@ksu.edu.sa

Abstract. In relational semantics, the input-output semantics of a program is a relation on its set of states. We generalize this in considering elements of Kleene algebras as semantical values. In a nondeterministic context, the *demonic* semantics is calculated by considering the worst behavior of the program. In this paper, we concentrate on while loops. *Calculating* the semantics of a loop is difficult, but *showing the correctness* of any candidate abstraction is much easier. For deterministic programs, Mills has described a checking method known as the *while statement verification rule*. A corresponding programming theorem for nondeterministic iterative constructs is proposed, proved and applied to an example. This theorem can be considered as a generalization of the while statement verification rule to nondeterministic loops.

Keywords: while loop, demonic semantics, relational abstraction, verification, Kleene algebra, rule, generalization.

1 Introduction

We use elements of Kleene algebras as abstractions of the input-output semantics of nondeterministic programs. In the concrete Kleene algebra of homogeneous binary relations, the operators $\cup$ and ; have been used for many years to define the so-called *angelic semantics*, which assumes that a program goes right when there is a possibility to go right. The *demonic choice* $\sqcup$ and *demonic composition* $\diamond$ do the opposite: if there is a possibility to go wrong, a program whose semantics is given by these operators goes wrong. The demonic semantics of a while loop is given as a fixed point of a monotonic function involving the demonic operators.

While there is no systematic way to calculate the relational abstraction of a while loop directly from the definition, it is possible to check the correctness of any candidate abstraction. For deterministic programs, Mills [15,16] has described a checking method known as the *while statement verification rule*. We generalize this rule to nondeterministic loops.

The rest of the paper is organized as follows. In Section 2, we present our mathematical tool, namely Kleene algebra [8]. There, a concept of type can

be defined that allows an abstract treatment of the domain (of definedness) of an element, and also of assertions. After some auxiliary results (Section 3 on fixed points and Section 4 on demonic operators), we present in Section 5 a generalization of the *while statement verification rule* of Mills. This is followed by an example of application in Section 6.

We note here that half of the generalized theorem has been shown by Sekerinski [20], who uses an approach based on predicative programming [12]. A related theorem has been given by Norvell [18] in the framework of predicative programming with time bounds. Norvell's theorem shows how to refine the specification R of a while loop under the condition that R is *strongly bounded*, which guarantees termination after a finite amount of time. Further refinement theorems for loops can be found in [2], presented in the framework of predicate transformers.

While the research programme around Mills's rule had originally been carried out by J. Desharnais and F. Tchier [21,22] in the framework of binary homogeneous relations, we show in the present paper that all the results can be generalized to the setting of Kleene algebras with types as introduced by B. Möller in [17]. In particular, the operator of forming the converse of an element is not needed. A remarkable feature is that the proofs in the generalized setting are considerably simpler and more perspicuous than the corresponding ones in terms of relations or predicate transformers.

2 Kleene Algebras

2.1 Definition and Basic Laws

In our definition of a Kleene algebra, we follow [8], since we want to admit general recursive definitions, not just the Kleene star. We are well aware that there are different definitions (see e.g. [13]).

Definition 2.1. *A* Kleene algebra (KA) *is a quintuple* $(K, \leq, \top, \cdot, 0, 1)$ *satisfying the following properties:*

(a) $(K, \leq)$ *is a complete lattice with least element* 0 *and greatest element* $\top$*. The supremum of a subset* $L \subseteq K$ *is denoted by* $\sqcup L$*.*

(b) $(K, \cdot, 1)$ *is a monoid.*

(c) The operation $\cdot$ *is universally disjunctive (i.e. distributes through arbitrary suprema) in both arguments.*

The supremum of two elements $x, y \in K$ is given by $x + y \triangleq \sqcup \{x, y\}$.

Perhaps the best-known example of a KA is LAN $\triangleq (\mathcal{P}(A^*), \subseteq, A^*, \bullet, \emptyset, \varepsilon)$, the algebra of formal languages over some alphabet A, where A^* is the set of all finite words over A, $\bullet$ denotes concatenation and ε the empty word (as usual, we identify a singleton language with its only element).

Another important KA is REL $\triangleq (\mathcal{P}(M \times M), \subseteq, M \times M, \ ;, \ \emptyset, \ I)$, the algebra of homogeneous binary relations over some set M under relational composition ; . More generally than the concrete relation algebra REL, every abstract relation algebra (see e.g. [7,19]) is a KA.

Definition 2.2. *A KA is called* Boolean *if its underlying lattice $(K, \leq)$ is a Boolean algebra, i.e. a completely distributive and complemented lattice. The complement of an element $a \in K$ is denoted by $\overline{a}$.*

2.2 Types

Definition 2.3. *A* type *of a KA is an element t with $t \leq 1$.*

This definition is best illustrated in the KA REL. There a type $1_T \subseteq I$ is a partial identity relation of the form $1_T \triangleq \{(x, x) \mid x \in T\}$ for some subset $T \subseteq M$. So it models the assertion of belonging to T.

Now, restriction of a relation $R \subseteq M \times M$ to arguments of type T, i.e. the relation $R \cap T \times M$, can also be described by composing R with 1_T from the left: $R \cap T \times M = 1_T ; R$. Similarly, co-restriction is modeled by composing a partial identity from the right. Finally, consider types $S, T \subseteq M$ and binary relation $R \subseteq M \times M$. Then $R \subseteq S \times T \Leftrightarrow 1_S ; R ; 1_T = R$. In other words, the "default typing" $M \times M$ of R can be narrowed down to $S \times T$ iff restriction to S and co-restriction to T do not change R.

These observations are the basis for our view of types as subidentities and our algebraic treatment of restriction and co-restriction. For a different, but related, approach see [13].

Lemma 2.4. *Assume a Boolean KA. Then the following hold:*

(a) All types are idempotent, i.e. $t \leq 1 \Rightarrow t \cdot t = t$.
(b) The infimum of two types is their product: $s, t \leq 1 \Rightarrow s \cdot t = s \sqcap t$. In particular, all types commute under the $\cdot$ operation.
(c) For all families L of types, $(\bigsqcap L) \cdot \top = \bigsqcap (L \cdot \top)$.

Definition 2.5. *The* negation *of a type $x \leq 1$ in a typed KA is $\neg x \triangleq \overline{x} \sqcap 1$.*

2.3 Domain and Codomain

Definition 2.6. *In a KA $(K, \leq, \top, \cdot, 0, 1)$, we define, for $a \in K$, the* domain $\ulcorner a$ *via the Galois connection (y ranges over types only!)* $\ulcorner a \leq y \overset{\text{def}}{\Leftrightarrow} a \leq y \cdot \top$.

This is well defined because of Lemma 2.4, see also [1]. Hence the operation $\ulcorner$ is universally disjunctive and therefore monotonic and strict. Moreover, the definition implies $a \leq \ulcorner a \cdot \top$. The *co-domain* $a\urcorner$ is defined symmetrically.

We list a number of useful properties of the domain operation (see again also [1]); analogous ones hold for the co-domain operation.

Lemma 2.7. *Consider a KA $(K, \leq, \top, \cdot, 0, 1)$ and $a, b, c \in K$.*

(a) $\ulcorner a = \min\{x : x \leq 1 \wedge x \cdot a = a\}$,
(b) $\ulcorner a \cdot a = a$,
(c) $x \leq 1 \wedge x \cdot a = a \Rightarrow \ulcorner a \leq x$,
(d) $\ulcorner (a \cdot b) \leq \ulcorner a$,
(e) $x \leq 1 \Rightarrow \ulcorner x = x$,
(f) $\ulcorner (a \cdot b) \leq \ulcorner (a \cdot \ulcorner b)$,
(g) $\ulcorner a = 0 \Leftrightarrow a = 0$.

According to Lemma 2.7(g) the domain of an element also decides its "definedness" if we identify 0 with $\bot$ as used in denotational semantics.

2.4 Locality of Composition

It should be noted that the converse inequation of Lemma 2.7(f) does not follow from our axiomatization. A counterexample is given in [10]. Its essence is that composition does not work "locally" in that only the domain of the right factor of a composition would decide about its definedness.

Therefore we say that a KA has *left-local composition* if it satisfies

$$\ulcorner b = \ulcorner c \Rightarrow \ulcorner(a \cdot b) = \ulcorner(a \cdot c) \ .$$

Right-locality is defined symmetrically. A KA has *local composition* if its composition is both left-local and right-local.

Lemma 2.8.

(a) A KA has left-local composition iff it satisfies $\ulcorner(a \cdot b) = \ulcorner(a \cdot \ulcorner b)$.
(b) If a KA has left-local composition then $\ulcorner(\ulcorner a \cdot b) = \ulcorner a \sqcap \ulcorner b = \ulcorner a \cdot \ulcorner b$.

Analogous properties hold for right-locality. In the sequel we only consider KAs with local composition. All examples given in Section 2.1 satisfy that property.

2.5 Type Implication

Types also play the rôle of assertions. The following operator will be instrumental in propagating assertions through compositions.

Definition 2.9. *The binary operator* $\rightarrow$*, called* type implication*, is defined as follows:*

$$a \rightarrow b \ \triangleq \ \neg\ulcorner(a \cdot \neg\ulcorner b) \ .$$

Hence $a \rightarrow b$ characterizes the type of points from which no computation as described by a may lead outside the domain of b. If a and b are types then a straightforward calculation shows that $a \rightarrow b = \neg a + b$, so that both the name "implication" and the symbol are justified.

This operator is closely related to the *monotype factor* as defined by Backhouse in [4]. For the case of a type t, one can interpret $a \rightarrow t$ also as $[a]t$, where $[a]$ is the modal "always" operator as used in dynamic logic (see, e.g., [11]).

Lemma 2.10. *Let t be a type.*

(a) $1 \rightarrow b = \ulcorner b$	
(b) $(a + b) \rightarrow c = (a \rightarrow c) \cdot (b \rightarrow c)$	
(c) $a \cdot b \rightarrow c = a \rightarrow (b \rightarrow c)$	*(Currying)*
(d) $(t \cdot a \rightarrow y) \cdot t = (a \rightarrow y) \cdot t$	*(Modus Ponens)*
(e) $(a \rightarrow t \cdot b) \cdot a = (a \rightarrow t \cdot b) \cdot a \cdot t$	*(Type Propagation)*
(f) $a \rightarrow b \cdot \ulcorner c = a \rightarrow b \cdot c$	*(Domain Absorption)*
(g) $a \rightarrow t \cdot b = (a \rightarrow t) \cdot (a \rightarrow b) = (a \rightarrow b) \cdot (a \rightarrow t)$	*(Weak Distributivity)*

3 Fixed Points

We recall a few basic facts about fixed points.

Definition 3.1. *A function f between complete lattices is* strict *if $f(0) = 0$ and* co-strict *if $f(\top) = \top$. Further, f is called* continuous *if $f(\sqcup L) = \sqcup f(L)$ for every chain L, and* co-continuous *iff $f(\sqcap L) = \sqcap f(L)$ for every chain L.*

Every universally disjunctive function is continuous and strict; every universally conjunctive function is co-continuous and co-strict. Moreover, every continuous or co-continuous function is monotonic.

Theorem 3.2. (Knaster/Tarski and Kleene)

(a) A monotonic function f on a complete semilattice has a least (greatest) fixed point $\mu(f)$ ($\nu(f)$) provided its set $\{x : x \geq f(x)\}$ of contracted elements (its set $\{x : x \leq f(x)\}$ of expanded elements) is non-empty. These fixed points satisfy

$$\begin{array}{rcl} \mu(f) & = & \sqcap(x :: f(x) = x)\} \;=\; \sqcap(x :: f(x) \leq x) \;, \\ \nu(f) & = & \sqcup(x :: f(x) = x)\} \;=\; \sqcup(x :: x \leq f(x)) \;. \end{array}$$

(b) Every monotonic function f on a complete lattice has a least fixed point $\mu(f)$ and a greatest fixed point $\nu(f)$.

(c) If f is continuous, then $\mu(f) = \sqcup(i : i \in \mathbb{N} : f^i(0))$. If f is co-continuous, then $\nu(f) = \sqcap(i : i \in \mathbb{N} : f^i(0))$.

Because we assume our KAs to be complete lattices (Definition 2.1), least and greatest fixed points of monotonic functions exist.

Definition 3.3. *Let $(X, \leq)$ be an ordered set. The relation $\ll$ on the set of functions on X is defined by $f \ll g \Leftrightarrow \forall(x : x \in X : f(x) \leq g(x))$.*

Theorem 3.4. (See, e.g., [3]) *The following properties hold:*

(a) $f \ll g \;\Rightarrow\; \mu(f) \leq \mu(g)$ (μ monotonic),

(b) Let g be continuous and strict. Then $f \circ g = g \circ h \;\Rightarrow\; \mu(f) = g(\mu(h))$ (μ-fusion law),

(c) $\mu(f \circ g) = f(\mu(g \circ f))$ (permutation law).

Analogous laws hold for the greatest fixed point. This can be shown more easily than by direct proof using the notion of a *dual* function.

Definition 3.5. *Let f be an function on a Boolean lattice. The* dual function *of f, notated $f^{\#}$, is defined by $f^{\#}(x) \triangleq \overline{f(\overline{x})}$.*

Lemma 3.6. *Let f be an function on a Boolean lattice. Then*

$$\nu(f^{\#}) = \overline{\mu(f)} \qquad \text{and} \qquad \mu(f^{\#}) = \overline{\nu(f)} \;.$$

¿From this and Theorem 3.4, one obtains

Theorem 3.7. *(a) $f \ll g \;\Rightarrow\; \nu(f) \leq \nu(g)$ (ν monotonic),*

(b) Let g be co-continuous and co-strict. Then $f \circ g = g \circ h \;\Rightarrow\; \nu(f) = g(\nu(h))$ (ν-fusion law),

(c) $\nu(f \circ g) = f(\nu(g \circ f))$ (permutation law).

3.1 Finite Iteration

Two central operations in KAs are *finite iteration* and *non-empty finite iteration*, defined for every element a by

$$a^* = \mu(x :: a \cdot x + 1) \qquad \text{and} \qquad a^+ = \mu(x :: a \cdot x + a) \ . \tag{1}$$

By the monotonicity of $+$ and $\cdot$, and the Knaster-Tarski-Kleene Theorem 3.2, these operations are well defined. Since $+$ and $\cdot$ are even continuous, we have

$$a^* = \bigsqcup(i : i \geq 0 : a^i) \qquad \text{and} \qquad a^+ = \bigsqcup(i : i > 0 : a^i) \ , \tag{2}$$

where $a^0 = 1$ and $a^{i+1} = a \cdot a^i$. This explains the name "finite iteration". In the algebra of relations, a^* coincides with the reflexive transitive closure and a^+ with the transitive closure.

3.2 Infinite Iteration and Termination

In the following, we introduce notions that are useful to describe the set of initial states of a program for which termination is guaranteed. These notions are the *infinite iteration* of an element, its *terminating part* and *progressive finiteness*.

Definition 3.8. *The* infinite iteration *of an element a is*

$$a^\omega \ \triangleq \ \nu(x :: a \cdot x) \ .$$

The terminating part *of an element a, denoted $\mathcal{T}(a)$ is the complement of the infinite iteration:*

$$\mathcal{T}(a) \ \triangleq \ \overline{a^\omega} \ = \ \mu(x :: \overline{a \cdot \overline{x}}) \quad \text{(by Lemma 3.6)} \ .$$

In the algebra of relations, the terminating part is also known as the *initial part* [19]. It is a vector characterizing the set of points s_0 such that there is no infinite chain $s_0, s_1, s_2, \ldots$, with $(s_i, s_{i+1}) \in a$, for all $i \geq 0$. In the semantics of while programs, an analogue of the terminating part will be used to represent sets of states from which no infinite looping is possible.

Definition 3.9. *An element a is said to be* progressively finite *iff $\mathcal{T}(a) = \top$ [19].*

In the algebra of relations, progressive finiteness of a relation R is the same as well-foundedness of $R^\smile$. By Boolean algebra, a is progressively finite iff $a^\omega = 0$.

We now list some useful properties of infinite iteration.

Theorem 3.10. *Let a and b be elements.*
(a) $a \leq b \Rightarrow a^\omega \leq b^\omega$,
(b) $a^* \cdot a^\omega = a^+ \cdot a^\omega = a^\omega$,
(c) $\neg\lceil(a^\omega) \cdot a$ *is progressively finite,*
(d) If b is progressively finite and $a \leq b$ then also a is progressively finite.

Proof. (a) Immediate from monotonicity of the fixed point operators.
(b) Easy calculation.
(c) Set $q \triangleq \neg\ulcorner(a^\omega) \cdot a$. Since $q \leq a$ we get $q^\omega \leq a^\omega$ and hence $\ulcorner(q^\omega) \leq \ulcorner(a^\omega)$. On the other hand, by Lemma 2.7(d,e)

$$\ulcorner(q^\omega) \;=\; \ulcorner(q \cdot q^\omega) \;=\; \ulcorner(\neg\ulcorner(a^\omega) \cdot a \cdot q^\omega) \;\leq\; \ulcorner(\neg\ulcorner(a^\omega)) \;=\; \neg\ulcorner(a^\omega) \;.$$

So $\ulcorner(q^\omega) \leq \ulcorner(a^\omega) \sqcap \neg\ulcorner(a^\omega) = 0$ and hence $q^\omega = 0$.
(d) Straightforward from (a).

Finally, we can give an analogue of the terminating part at the level of types:

Definition 3.11. $\mathcal{T}_\ulcorner(a) \triangleq \mu(x :: a \to x)$.

Using the correspondence with the modal operator, we have that $\mathcal{T}_\ulcorner(a) = \mu(x :: [a]x)$. In the propositional μ-calculus, this is known as the *halting predicate* (see, e.g., [11]).

It is easy to check that $\neg\ulcorner(a^\omega)$ is a fixed point of $(x :: a \to x)$. Hence,

Corollary 3.12. *(a)* $\mathcal{T}_\ulcorner(a) \leq \neg\ulcorner(a^\omega)$.
(b) $\mathcal{T}_\ulcorner(a) \cdot a$ *is progressively finite.*

Proof. (a) is immediate from the least fixed point property of $\mathcal{T}_\ulcorner(a)$. For (b), by monotonicity of $\_^\omega$ we get $(\mathcal{T}_\ulcorner(a) \cdot a)^\omega \leq (\neg\ulcorner(a^\omega) \cdot a)^\omega = 0$ using Theorem 3.10(c).

3.3 Connecting Finite and Infinite Iteration

The next theorem connects finite and infinite iteration.

Theorem 3.13. *Let* $f(x) \triangleq a \cdot x + b$. *Then* $\mu(f) = a^* \cdot b$ *and* $\nu(f) = a^* \cdot b + a^\omega$.

The proof is by fixed point fusion. For the second assertion, this depends crucially on the complete distributivity of the underlying lattice. The theorem entails the following corollary that highlights the importance of progressive finiteness in the simplification of fixed point-related properties.

Corollary 3.14. *Let again* $f(x) \triangleq a \cdot x + b$. *If* a *is progressively finite, then* f *has a unique fixed point, viz.* $a^* \cdot b$ *[3].*

To conclude this section, we study analogous iterations at the level of types.

Theorem 3.15. *Let* t *be a type and set, for* $x \in \mathrm{TYP}$, $h(x) \triangleq \ulcorner(a \cdot x) + t$ *and* $k(x) \triangleq \neg t \cdot (a \to x)$.
(a) $x \leq 1 \Rightarrow h(x) = \neg k(\neg x)$,
(b) $\mu(h) = \ulcorner(a^* \cdot t)$,
(c) $\nu(h) = \ulcorner(a^* \cdot t) + \neg\mathcal{T}_\ulcorner(a)$,
(d) $\mathcal{T}_\ulcorner(a) \leq \ulcorner(a^* \cdot \neg\ulcorner a)$,
(e) $\mu(k) = \mathcal{T}_\ulcorner(a) \cdot (a^* \to \neg t)$,
(f) $\nu(k) = (a^* \to \neg t)$.

Proof. Part (a) is straightforward. Parts (b) and (e) follow by fixed point fusion, which is applicable by continuity and strictness of the $\ulcorner\_$ and $\cdot$ operations. Note that $h(x) \leq 1$ and $k(x) \leq 1$ for any $x \in K$. Hence, any fixed point of h or k is a type. Because the types constitute a complete Boolean algebra, one can consider h and k to be functions on the set of types for the purpose of calculating fixed points. Thus, h and k are dual (Part (a)). One can then apply Lemma 3.6 to obtain Parts (c) and (f). For assertion (d), we first show $(a \rightarrow x) \leq \ulcorner(a \cdot x) + \neg\ulcorner a$; the proof uses shunting, locality of composition, distributivity and Boolean algebra:

$$(a \rightarrow x) \leq \ulcorner(a \cdot x) + \neg\ulcorner a \;\Leftrightarrow\; \neg\ulcorner(a \cdot \neg\ulcorner x) \leq \ulcorner(a \cdot x) + \neg\ulcorner a \;\Leftrightarrow\; \ulcorner a \leq \ulcorner(a \cdot \ulcorner x) + \ulcorner(a \cdot \neg\ulcorner x) \;\Leftrightarrow\; \ulcorner a \leq \ulcorner a \;.$$

Now, using the above derivation, the monotonicity of μ, and (b), we can conclude

$$\mathcal{T}_{\ulcorner}(a) \;=\; \mu(x :: a \rightarrow x) \;\leq\; \mu(x :: \ulcorner(a \cdot x) + \neg\ulcorner a) \;=\; \ulcorner(a^* \cdot \neg\ulcorner a) \;.$$

4 The Demonic Operators

4.1 Refinement Ordering

We now define a partial ordering, called the *refinement ordering*. This ordering induces a complete join semilattice, called a *demonic semilattice*. The associated operations are demonic join ($\sqcup\!\!\!\sqcup$), demonic meet ($\sqcap\!\!\!\sqcap$) and demonic composition ($\diamond$). Again, we generalize from the case of relation algebra to general KAs. For more details on relational demonic semantics and demonic operators, see [4,5,6,7,9,21].

Definition 4.1. *We say that an element a* refines *an element b [14], denoted by $a \sqsubseteq\!\!\!\sqsubseteq b$, iff* $\ulcorner b \leq \ulcorner a \;\wedge\; \ulcorner b \cdot a \leq b$.

It is easy to show that $\sqsubseteq\!\!\!\sqsubseteq$ is indeed a partial ordering.

Theorem 4.2. *(a) The partial order $\sqsubseteq\!\!\!\sqsubseteq$ induces a complete upper semilattice, ie., every subset $L \subseteq K$ has a least upper bound (wrt $\sqsubseteq\!\!\!\sqsubseteq$) $\bigsqcup L$ which is called its* demonic join. *One has*

$$\bigsqcup L \;=\; (\bigsqcap(a : a \in L : \ulcorner a)) \cdot \bigsqcup L \qquad \text{with} \qquad \ulcorner(\bigsqcup L) \;=\; \bigsqcap(a : a \in L : \ulcorner a) \;.$$

(b) $a \sqsubseteq\!\!\!\sqsubseteq b \;\Leftrightarrow\; a \sqcup\!\!\!\sqcup b = b$.

(c) If a and b satisfy the condition $\ulcorner(a \sqcap b) = \ulcorner a \sqcap \ulcorner b$, their greatest lower bound, denoted $a \sqcap\!\!\!\sqcap b$ and called their demonic meet, *exists and its value is*

$$a \sqcap\!\!\!\sqcap b \;=\; (a \sqcap b) + \neg\ulcorner a \cdot b + \neg\ulcorner b \cdot a \qquad \text{with} \qquad \ulcorner(a \sqcap\!\!\!\sqcap b) = \ulcorner a + \ulcorner b \;.$$

Otherwise, the greatest lower bound does not exist.

The existence condition for $\sqcap\!\!\!\sqcap$ simply means that on the intersection of their domains, a and b have to agree for at least one value.

4.2 Demonic Composition

Definition 4.3. *Let a and b be elements. The* demonic composition *of a and b, denoted by $a \,\square\, b$, is defined as* $a \,\square\, b \triangleq (a \rightarrow b) \cdot a \cdot b$.

In the algebra of relations, a pair (s,t) belongs to $a \,\square\, b$ if and only if it belongs to $a \cdot b$ and there is no possibility of reaching, from s, by a, an element u that does not belong to the domain of b. For example, with $a \triangleq \{(0,0),(0,1),(1,2)\}$ and $b \triangleq \{(0,0),(2,3)\}$, one finds that $a \,\square\, b = \{(1,3)\}$; the pair $(0,0)$, which belongs to $a \cdot b$, does not belong to $a \,\square\, b$, since $(0,1) \in a$ and 1 is not in the domain of b. Note that we assign to $\square$ and $\cdot$ the same binding power.

A fundamental property is

Theorem 4.4. *The demonic composition is associative.*

For the next theorem we need a notion of determinacy [10].

Definition 4.5. *An element a is* deterministic *iff* $\mathrm{CD}(a)$ *holds, where*

$$\mathrm{CD}(a) \triangleq \forall(b : b \leq a : b = \ulcorner b \cdot a) \qquad \textit{(characterization by domain).}$$

We quote from [10]:

Lemma 4.6. *All types are deterministic. If a is deterministic and $b \leq a$, then b is deterministic as well.*

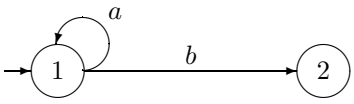
The following properties are shown by straightforward calculations.

Theorem 4.7. *(a) a deterministic* $\Rightarrow a \,\square\, b = a \cdot b$,
(b) $\ulcorner a \cdot \ulcorner b = 0 \Rightarrow (a+b) \,\square\, c = a \,\square\, c + b \,\square\, c$,
(c) $\ulcorner a \cdot \ulcorner b = 0 \Rightarrow a \sqcap b = a + b$.

5 The Semantics of Nondeterministic Loops

5.1 Intuition and Notation

A general nondeterministic loop is best described by a graph of the form



It may "execute" a as long as the intermediate states remain in the domain of a and it may exit if a state in the domain of b is reached. The domains of a and b need not be disjoint. Since a may be nondeterministic, it can take a starting state s to many successor states. If among these there exists a state outside the domains of a and b (abnormal termination), then in the demonic view s must be excluded from the domain of the loop semantics. Hence, in addition to $\mathcal{T}_{\ulcorner}(a)$, we introduce a type $\mathcal{P}(a,b)$ ($\mathcal{P}$ stands for *p*roper) that characterizes the states from which no abnormal termination is possible.

We now define the corresponding semantic functions formally. Let a and b be elements. The abbreviations φ, $\varphi_\ulcorner, \mathcal{P}(a,b), s_\mu$ and s_ν are defined as follows:

$$\begin{array}{rcl@{\qquad\qquad}rcl} \varphi(x) & \triangleq & (a \to x) \cdot (a \cdot x + b) \ , & \mathcal{P}(a,b) & \triangleq & a^* \to (a+b) \ , \\ \varphi_\ulcorner(x) & \triangleq & (a \to x) \cdot \ulcorner(a+b) \ , & s_\mu & \triangleq & \mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a) \cdot a^* \cdot b \ . \end{array} \tag{3}$$

The subscript $\ulcorner$ in $\varphi_\ulcorner$ expresses that $\varphi_\ulcorner$ is a domain-oriented counterpart of φ.

The element s_μ, *which we take as the semantics of the loop*, is the restriction of the angelic loop semantics a^*b to $\mathcal{P}(a,b)$ and $\mathcal{T}_\ulcorner(a)$. Hence the domain of s_μ represents the set of states from which proper termination is guaranteed. We want to show that s_μ is the least fixed point of φ.

Lemma 5.1. *Assume, as in Theorem 3.13,* $f(x) \triangleq a \cdot x + b$.
(a) If s is a type then $\varphi(s \cdot x) = \varphi_\ulcorner(s \cdot x) \cdot f(x)$.
(b) $\varphi(x) = (a \to x) \cdot f(x) = \varphi_\ulcorner(x) \cdot f(x) = a \,\square\, x + (a \to x) \cdot b$.
(c) If $\ulcorner a \cdot \ulcorner b = 0$, *then* $\varphi(x) = a \,\square\, x \sqcap b$.

Proof. (a)
$$\begin{array}{ll} & \varphi(s \cdot x) \\ = & \quad \{\![\text{ definitions }]\!\} \\ & (a \to s \cdot x) \cdot (a \cdot s \cdot x + b) \\ = & \quad \{\![\text{ distributivity and type propagation (Lemma 2.10(e)) }]\!\} \\ & (a \to s \cdot x) \cdot (a \cdot x + b) \\ = & \quad \{\![\text{ since } \ulcorner(a \cdot x + b) \le \ulcorner(a+b) \ ]\!\} \\ & (a \to s \cdot x) \cdot \ulcorner(a+b) \cdot (a \cdot x + b) \\ = & \quad \{\![\text{ definitions }]\!\} \\ & \varphi_\ulcorner(s \cdot x) \cdot f(x) \end{array}$$

(b) Immediate from (a) by setting $s = 1$ and from the definitions.
(c) This follows from Part (b) by monotonicity of $\ulcorner$, Boolean laws and Theorem 4.7(c).

Item (c) justifies why we are talking about a demonic star operator.

5.2 Properties of the Semantics

The following lemma presents the relationship between the fixed points of the functions φ and $\varphi_\ulcorner$ (Equations 3). It is again proved by straightforward calculation.

Lemma 5.2. *(a)* $\ulcorner(\varphi(x)) = \varphi_\ulcorner(\ulcorner x)$,
(b) If y is a fixed point of φ then $\ulcorner y$ is a fixed point of $\varphi_\ulcorner$.

In the following we give bounds on the fixed points of $\varphi_\ulcorner$.

Theorem 5.3. *(a)* $\mu(\varphi_\ulcorner) = \mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a)$ *and* $\nu(\varphi_\ulcorner) = \mathcal{P}(a,b)$.
(b) If y is a fixed point of φ, then $\mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner a \ \le \ \ulcorner y \ \le \ \mathcal{P}(a,b)$.

Proof. Immediate from Lemma 5.2(b) and Theorem 3.15(e,f).

The next theorem characterizes the domain of s_μ. It is the set of points for which normal termination is guaranteed (no possibility of abnormal termination or infinite loop).

Theorem 5.4. *(a)* $\mathcal{P}(a,b) \;\leq\; \ulcorner(a^* \cdot b) + \neg\mathcal{T}_\ulcorner(a)$,
(b) $\mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a) \;\leq\; \mathcal{T}_\ulcorner(a) \cdot \ulcorner(a^* \cdot b)$,
(c) $\ulcorner s_\mu \;=\; \mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a) \;=\; \mu(\varphi_\ulcorner)$.

Proof. (a) $\mathcal{P}(a,b) \leq \ulcorner(a^* \cdot b) + \neg\mathcal{T}_\ulcorner(a)$
$\Leftrightarrow$ {[shunting]}
$\mathcal{T}_\ulcorner(a) \leq \ulcorner(a^* \cdot b) + \neg\mathcal{P}(a,b)$
$\Leftrightarrow$ {[Equations 3, Definition 2.9, Boolean algebra, Lemma 2.8]}
$\mathcal{T}_\ulcorner(a) \leq \ulcorner(a^* \cdot \ulcorner b) + \ulcorner(a^* \cdot \neg\ulcorner a \cdot \neg\ulcorner b)$
$\Leftrightarrow$ {[distributivity, Boolean algebra]}
$\mathcal{T}_\ulcorner(a) \leq \ulcorner(a^* \cdot (\ulcorner b + \neg\ulcorner a))$
$\Leftarrow$ {[monotonicity]}
$\mathcal{T}_\ulcorner(a) \leq \ulcorner(a^* \cdot \neg\ulcorner a)$
$\Leftrightarrow$ {[Theorem 3.15(d)]}
true

(b) Immediate from (a).

(c) $\ulcorner s_\mu$
$=$ {[by (3)]}
$\ulcorner(\mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a) \cdot a^* \cdot b)$
$=$ {[locality of composition (Lemma 2.8)]}
$\mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a) \cdot \ulcorner(a^* \cdot b)$
$=$ {[(b), Boolean algebra]}
$\mathcal{P}(a,b) \cdot \mathcal{T}_\ulcorner(a)$

In the following theorem, we show that s_μ is a fixed point of φ.

Theorem 5.5. $\varphi(s_\mu) = s_\mu$.

Proof. $\varphi(s_\mu)$
$=$ {[Equations 3, and Theorems 5.4(c) and 3.13]}
$\varphi(\mu(\varphi_\ulcorner) \cdot \mu(f))$
$=$ {[Lemma 5.1(a)]}
$\varphi_\ulcorner(\mu(\varphi_\ulcorner) \cdot \mu(f)) \cdot f(\mu(f))$
$=$ {[Equations 3, Theorems 5.4(c) and 3.13, fixed point property]}
$\varphi_\ulcorner(s_\mu) \cdot \mu(f)$
$=$ {[Equations 3, Definition 2.9, Lemma 2.7]}
$\varphi_\ulcorner(\ulcorner s_\mu) \cdot \mu(f)$
$=$ {[Theorem 5.4(c)]}
$\varphi_\ulcorner(\mu(\varphi_\ulcorner)) \cdot \mu(f)$
$=$ {[fixed point property]}

$$
\begin{array}{ll}
 & \mu(\varphi_\Gamma) \cdot \mu(f) \\
= & \quad \{\![\text{ Equations 3, Theorems 5.4(c) and 3.13 }]\!\} \\
 & s_\mu
\end{array}
$$

The following theorem uniquely characterizes the least fixed point of φ by a simple condition and shows that s_μ is the least fixed point of φ.

Theorem 5.6. *Recall Equations 3. For all elements a,*

(a) $c = \mu(\varphi) \;\Leftrightarrow\; \varphi(c) = c \wedge \ulcorner c \leq \mathcal{T}_\Gamma(a)$,
(b) $\mu(\varphi) = s_\mu$.

Proof. (a) ($\Rightarrow$) Assume $c = \mu(\varphi)$. The property $\varphi(c) = c$ then obviously follows. From Theorem 5.5 and Theorem 3.2(a), we get $c \leq s_\mu$ and hence, by Theorem 5.4(c), $\ulcorner c \leq \ulcorner s_\mu = \mathcal{P}(a,b) \cdot \mathcal{T}_\Gamma(a) \leq \mathcal{T}_\Gamma(a)$.

($\Leftarrow$) Assume $\varphi(c) = c$ and $\ulcorner c \leq \mathcal{T}_\Gamma(a)$. Theorem 5.3 implies $\mathcal{P}(a,b) \cdot \mathcal{T}_\Gamma(a) \leq \ulcorner c \leq \mathcal{P}(a,b)$. Hence $\ulcorner c = \mathcal{P}(a,b) \cdot \mathcal{T}_\Gamma(a) = \mu(\varphi_\Gamma)$. This is used in the following derivation, which also employs Lemma 5.1(b), domain absorption (Lemma 2.10(f)), a fixed point property and distributivity.

$$
\begin{array}{l}
c \;=\; \varphi(c) \;=\; \varphi_\Gamma(c) \cdot f(c) \;=\; \varphi_\Gamma(\ulcorner c) \cdot f(c) \;=\; \varphi_\Gamma(\mu(\varphi_\Gamma)) \cdot f(c) \;= \\
\quad \mu(\varphi_\Gamma) \cdot f(c) \;=\; \mu(\varphi_\Gamma) \cdot a \cdot c + \mu(\varphi_\Gamma) \cdot b \;.
\end{array}
$$

By Theorem 5.4(c), Corollary 3.12(b) and Theorem 3.10(d), $\mu(\varphi_\Gamma) \cdot a$ is progressively finite. Invoking Corollary 3.14 shows that the function $(x :: \mu(\varphi_\Gamma) \cdot a \cdot x + \mu(\varphi_\Gamma) \cdot b)$ has a unique fixed point. Thus all elements c such that $\varphi(c) = c$ and $\ulcorner c \leq \mathcal{T}_\Gamma(a)$ are equal. But $\mu(\varphi)$ is such an element, as we have shown above (part $\Rightarrow$). We conclude that $c = \mu(\varphi)$.

(b) By Theorem 5.5, $s_\mu = \varphi(s_\mu)$. By Theorem 5.4(c), $\ulcorner s_\mu \leq \mathcal{T}_\Gamma(a)$. Now the claim is a consequence of part (a) of this theorem.

5.3 Relating Angelic and Demonic Semantics

In the following, we will show that the element s_μ is the greatest fixed point with respect to $\mathbb{E}$ of the function φ (Equations 3). But first, we show

Lemma 5.7. *The function φ is monotonic wrt $\mathbb{E}$.*

Proof. Assume $x \mathbb{E} y$. First, using Lemma 5.2(a) and monotonicity of φ_Γ wrt $\leq$,

$$
\ulcorner\varphi(y) \leq \ulcorner\varphi(x) \;\Leftrightarrow\; \varphi_\Gamma(\ulcorner y) \leq \varphi_\Gamma(\ulcorner x) \;\Leftarrow\; \ulcorner y \leq \ulcorner x \;.
$$

Second, using Lemma 5.2(a), Equations 3, Lemma 2.10(g), the hypothesis and monotonicity of φ wrt $\leq$,

$$
\begin{array}{l}
\ulcorner\varphi(y) \cdot \varphi(x) \;=\; \varphi_\Gamma(\ulcorner y) \cdot \varphi(x) \;=\; (a \rightarrow \ulcorner y) \cdot \ulcorner(a+b) \cdot (a \rightarrow x) \cdot (a \cdot x + b) \;= \\
(a \rightarrow \ulcorner y \cdot x) \cdot (a \cdot \ulcorner y \cdot x + b) \;=\; \varphi(\ulcorner y \cdot x) \;\leq\; \varphi(y) \;.
\end{array}
$$

Recall that $\mathbb{L}$ is a complete $\sqcup$-semilattice. Since the function φ is $\mathbb{L}$-monotonic and s_μ is a fixed point, i.e. an expanded element, by Theorem 3.2 the $\mathbb{L}$-greatest fixed point of the function φ exists and is given by

$$w = \bigsqcup(x :: x = \varphi(x)) \ . \tag{4}$$

Theorem 5.8. *The element s_μ (Equations 3) is the $\mathbb{L}$-greatest fixed point of φ, that is $s_\mu = \nu_{\mathbb{L}}(\varphi)$.*

Proof. Since $s_\mu = \varphi(s_\mu)$, by (4) we get $s_\mu \sqsubseteq w$. Hence, the definition of $\mathbb{L}$ implies $\ulcorner w \leq \ulcorner s_\mu$. Using Theorem 5.4(c) now gives $\ulcorner w \leq \mathcal{T}_\ulcorner(a)$, so that, by Theorem 5.6(a), we finally obtain $w = \mu_\leq(\varphi) = s_\mu$.

In other words, the least fixed point of φ wrt $\leq$ is equal to the greatest fixed point of the same function φ wrt $\mathbb{L}$.

6 Application

In Mills's approach, the semantics w of a deterministic loop **do** $\mathtt{g} \to \mathtt{C}$ **od** is given as the least fixed point (wrt $\leq$) of the function

$$w_{gc}(x) \ \triangleq \ g \cdot c \cdot x + \neg g, \tag{5}$$

where the type g is the semantics of the loop guard $\mathtt{g}$ and the element c is the semantics of the loop body $\mathtt{C}$.

Lemma 6.1. *If the loop body c is deterministic, then*

$$w_{gc}(x) \ = \ (g \cdot c \to x) \cdot (g \cdot c \cdot x + \neg g) \ = \ g \,\square\, c \,\square\, x \ \sqcap \ \neg g \ .$$

Hence, in this case, the demonic and angelic semantics coincide, as expected. In this case, one can also prove that, under mild additional assumptions on the underlying KA, the semantics of the loop is deterministic as well.

Calculating the relational abstraction (semantics) of a loop is difficult, but *showing the correctness* of any candidate abstraction is much easier. For deterministic programs, Mills [15,16] has described a checking method known as the *while statement verification rule*. In a nondeterministic context, the abstraction is calculated by considering the worst behavior of the program (*demonic semantics*) [21]. Given a loop condition and a loop body, Theorem 5.6 (with $a \triangleq g \cdot c$ and $b \triangleq \neg g$; notice that $\ulcorner a \cdot \ulcorner b = 0$) can be used to verify if an element w is indeed the semantics of the loop.

The following example is rather contrived, but it is simple and fully illustrates the various cases that may happen. Consider the following loop, where the unique variable n ranges over the set of integers [7,22]:

Example 6.2. Consider the program

$$\begin{array}{l}\textbf{do } \texttt{n} > 0 \to \textbf{if } \texttt{n} = 1 \to \texttt{n} := 1 \quad [] \ \texttt{n} = 1 \to \texttt{n} := -3 \\ \qquad\qquad\quad [] \ \texttt{n} = 3 \to \texttt{n} := 2 \quad [] \ \texttt{n} = 3 \to \texttt{n} := -1 \\ \qquad\qquad\quad [] \ \texttt{n} \geq 4 \to \texttt{n} := -4 \\ \qquad\qquad\quad \textbf{fi} \quad \textbf{od}\end{array}$$

Notice that all $n > 0$ such that $n \bmod 4 = 1$ may lead to termination with a final value $n' = -3$, but may also lead to an infinite loop over the value $n = 1$; these initial values of n do not belong to the domain of the element giving the semantics of the loop. Note also that all $n > 0$ such that $n \bmod 4 = 3$ may lead to termination with a final value $n' = -1$, but may also lead to a value $n = 2$, for which the loop body is not defined (by the semantics of **if fi**); these n do not belong to the domain of w. Because they also lead to $n = 2$, all $n > 0$ such that $n \bmod 4 = 2$ do not belong to the domain of w.

The semantics of the loop guard in the concrete KA REL is given by:

$$g = \{n > 0 \wedge n' = n\} \qquad (\text{whence } \neg g = \{n \leq 0 \wedge n' = n\}).$$

The semantics of the loop body is:

$$\begin{array}{rl} c \ = & \{n = 1 \wedge n' = n\} \mathbin{\triangledown} (\{n' = 1\} \sqcup \{n' = -3\}) \\ & \sqcap \{n = 3 \wedge n' = n\} \mathbin{\triangledown} (\{n' = 2\} \sqcup \{n' = -1\}) \\ & \sqcap \{n \geq 4 \wedge n' = n\} \mathbin{\triangledown} \{n' = n - 4\} \ . \end{array}$$

By Proposition 4.7(a), $g \mathbin{\triangledown} c = g \cdot c = c$. Using Theorem 5.6(a), we show that

$$w \ \triangleq \ \{(n \leq 0 \wedge n' = n) \ \vee \ (n > 0 \wedge n \bmod 4 = 0 \wedge n' = 0)\}$$

is the semantics of the loop. The condition $\varphi(w) = w$ of theorem 5.6(a) follows from straightforward calculations. The second condition $\ulcorner w \leq \mathcal{T}_\ulcorner(g \cdot c)$ can be established informally by noting that the domain of w is $\{n \leq 0 \vee n \bmod 4 = 0\}$, and that there is no infinite sequence by $g \cdot c$ for any n in the domain of w.

A more satisfactory way to show $\ulcorner w \leq \mathcal{T}_\ulcorner(g \cdot c)$ is to calculate $\mathcal{T}_\ulcorner(g \cdot c)$. However, because $\mathcal{T}_\ulcorner(g \cdot c)$ characterizes the domain of guaranteed termination of the associated loop, there is no systematic way to compute it (this would solve the halting problem). To demonstrate termination of the loop from every state in the domain of w, classical proofs based on variant functions or well-founded sets could be given. But formal arguments based on the definition of the terminating part (Definition 3.8) can also be used [7].

In this example, Theorem 5.6 was used to *verify* that the guessed semantics w of the loop was correct, given the semantics g of the loop guard and c of the loop body. The theorem can also be used in the other direction. If we are given a specification w, we can guess g and c, and then apply Theorem 5.6 to verify the correctness of the guess. If it is correct, then a loop of the form **do** $\texttt{g} \to \texttt{C}$ **od**, where `C` is an implementation of c, is correct with respect to w.

Acknowledgments. The authors thank Thorsten Ehm and Dexter Kozen for helpful comments. This research was supported by FCAR (Fonds pour la Formation de Chercheurs et l'Aide à la Recherche, Québec) and NSERC (Natural Sciences and Engineering Research Council of Canada).

References

1. C. J. Aarts. Galois connections presented calculationally. Eindhoven University of Technology, Dept. of Mathematics and Computer Science, July 1992.
2. R. Back and J. von Wright. *Refinement Calculus — A Systematic Introduction*. Springer, 1998.
3. R. C. Backhouse *et al*. Fixed point calculus. *Inform. Proc. Letters*, 53:131–136, 1995.
4. R. C. Backhouse and J. van der Woude. Demonic operators and monotype factors. *Mathematical Structures in Comput. Sci.*, 3(4):417–433, 1993.
5. R. Berghammer and H. Zierer. Relational Algebraic semantics of deterministic and nondeterministic programs. *Theoret. Comput. Sci.*, 43:123–147, 1986.
6. N. Boudriga, F. Elloumi, and A. Mili, On the lattice of specifications: Applications to a specification methodology. *Formal Aspects of Computing*, 4:544–571, 1992.
7. C. Brink, W. Kahl, and G. Schmidt (eds). *Relational Methods in Computer Science*. Springer, 1997.
8. J. H. Conway. *Regular Algebra and Finite Machines*. Chapman and Hall, London, 1971.
9. J. Desharnais, N. Belkhiter, S. B. M. Sghaier, F. Tchier, A. Jaoua, A. Mili, and N. Zaguia. Embedding a demonic semilattice in a relation algebra. *Theoret. Comput. Sci.*, 149(2):333–360, 1995.
10. J. Desharnais and B. Möller. Characterizing functions in Kleene algebras. In J. Desharnais (ed.), *Proc. 5th Seminar on Relational Methods in Computer Science (RelMiCS'5)*. Université Laval, Canada, pages 55–64, 2000.
11. D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. Forthcoming book.
12. E. Hehner. Predicative programming, Parts I and II. *CACM*, 27:134–151, 1984.
13. D. Kozen: Kleene algebras with tests. *ACM TOPLAS* 19:427–443, 1997.
14. A. Mili, J. Desharnais, and F. Mili. Relational heuristics for the design of deterministic programs. *Acta Inform.*, 24(3):239–276, 1987.
15. H. D. Mills. The new math of computer programming. *CACM*, 18(1):43–48, 1975.
16. H. D. Mills, V. R. Basili, J. D. Gannon and R. G. Hamlet. *Principles of Computer Programming. A Mathematical Approach.* Allyn and Bacon, Inc., 1987.
17. B. Möller: Typed Kleene algebras. Universität Augsburg, Institut für Informatik, Report, 1999
18. T. S. Norvell. Predicative semantics of loops. In R. S. Bird and L. Meertens (eds), *Algorithmic Languages and Calculi*, Chapman & Hall, 1997, pages 415–437.
19. G. Schmidt and T. Ströhlein. *Relations and Graphs*. EATCS Monographs in Computer Science, Springer-Verlag, Berlin, 1993.
20. E. Sekerinski. A calculus for predicative programming. *Second International Conf. on the Mathematics of Program Construction*. R. S. Bird, C. C. Morgan and J. C. P. Woodcock (eds), Oxford, June 1992, *Lect. Notes in Comput. Sci.*, Vol. 669, Springer-Verlag, 1993.
21. F. Tchier. Sémantiques relationelles demoniaques et vérification de boucles nondéterministes. Ph.D. Thesis, Département de Mathématiques, Université Laval, Canada, 1996.
22. F. Tchier and J. Desharnais. Applying a generalization of a theorem of Mills to generalized looping structures. Colloquium *Science and Engineering for Software Development*, organized in the memory of Dr. Harlan D. Mills, and affiliated to the *21st International Conference on Software Engineering*, Los Angeles, 18 May 1999, pages 31–38.