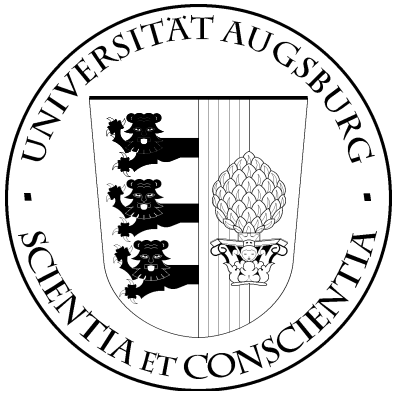
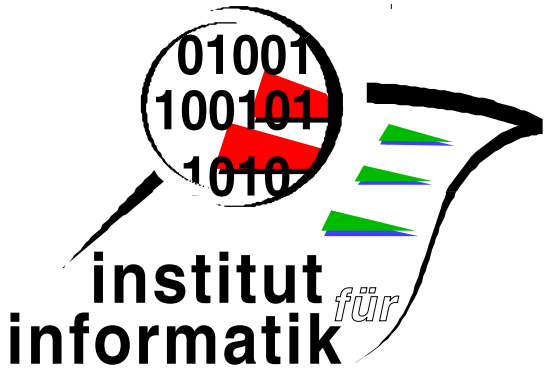
Universität Augsburg



Typed Kleene Algebras

Bernhard Möller

Report 1999-8 Dezember 1999



Institut für Informatik
D-86135 Augsburg

Typed Kleene Algebras

Bernhard Möller

Institut für Informatik, Universität Augsburg, D-86135 Augsburg, Germany

1 Introduction

Kleene algebras provide a convenient and powerful algebraic axiomatisation of a complete lattice that is endowed with a sequential composition operation. Models include formal languages under concatenation, relations under standard composition, sets of graph paths under path concatenation and sets of streams under concatenation. The least and greatest fixpoint operators of a complete lattice allow definitions of the finite and infinite iteration operators * and $^\omega$, resp.

The abstract setting of Kleene algebras was used in [3] for the schematic derivation of a class of algorithms that abstracts layer-oriented graph traversals such as breadth-first search. It was also shown that the standard efficiency improvement for such algorithms, viz. carrying along a set of already visited vertices, can completely be transferred to the abstract level. The set of already visited vertices is used to restrict the underlying graph appropriately. So we need abstract counterparts of the notions of (sub)sets and restriction. The former role is taken over by elements of the Kleene algebra that are called *types*, whereas restriction can be modeled, as in the case of relations, by composition with a type.

The paper provides the corresponding definitions and proves a number of useful properties of types and restriction. In particular, we give algebraic characterisations of the domain and codomain operations. They are presented in such a way that they can also be used for "one-sided" Kleene algebras. An instance of this is presented by the algebra of streams; there the domain of a set S of streams is the set of first letters that occur in streams in S, whereas a corresponding co-domain operation does not make sense if infinite streams occur in the set. This treatment fits well with systems such as R. Dijkstra's computation calculus [5].

We also relate the theory to the abstract treatment of assertions presented in [11].

2 Kleene Algebras

A *left Kleene algebra (LKA)* is a quintuple $(S, \Sigma, \cdot, 0, 1)$ consisting of a set S, operations $\Sigma : \mathcal{P}(S) \to S$ and $\cdot : S \times S \to S$ as well as elements $0, 1 \in S$ satisfying the following properties:

1.

$$\begin{array}{rcll} \Sigma\,\emptyset & = & 0 & , \\ \Sigma\,\{x\} & = & x & (x \in S) \ , \\ \Sigma(\cup\mathcal{K}) & = & \Sigma\,\{\Sigma\,K : K \in \mathcal{K}\} & (\mathcal{K} \subseteq \mathcal{P}(S)) \ . \end{array}$$

Then one can define a partial order as follows:

$$x \leq y \stackrel{\text{def}}{\Leftrightarrow} x + y = y \ , \tag{1}$$

where

$$x + y \stackrel{\text{def}}{=} \Sigma\{x, y\} \ . \tag{2}$$

Then $(S, \leq)$ forms a complete lattice in which Σ coincides with the supremum operator. We denote the greatest element of that lattice by $\top$. Moreover, $+$ as the binary supremum operator is associative, commutative and idempotent.

2. The element 1 is left-neutral w.r.t. the $\cdot$ operation. Moreover, $\cdot$ is associative and $\leq$-monotonic in its right argument.
3. The operation $\cdot$ is universally disjunctive (and hence also $\leq$-monotonic) in its left argument, ie. for all $K \subseteq S$ and $x \in S$ we have

$$(\Sigma\, K) \cdot x \ = \ \Sigma\, (K \cdot x) \ ,$$

where $\cdot$ at the rhs of the latter equation is the pointwise extension of the original $\cdot$ operation, ie. $K \cdot x \stackrel{\text{def}}{=} \{y \cdot x \mid y \in K\}$.

By postulates 3 and 1, $\cdot$ is left-strict w.r.t. 0:

$$0 \cdot x \ = \ 0 \ .$$

So 0 is the least element w.r.t. $\leq$. Another easy consequence of the postulates is

Corollary 21 $\top \cdot \top \ = \ \top$.

Proof:

$$\begin{array}{ll} & \top \\ = & \quad \{\![\text{ neutrality }]\!\} \\ & 1 \cdot \top \\ \leq & \quad \{\![\text{ monotonicity }]\!\} \\ & \top \cdot \top \ . \end{array}$$

The converse inequation is trivial. ■

For a binary operation $\cdot : M \times M \ \to \ M$ we define its *mirror operation* $\breve{\cdot} : M \times M \ \to \ M$ by

$$x \breve{\cdot} y \ = \ y \cdot x \ .$$

We call a quintuple $(S, \Sigma, \cdot, 0, 1)$ a *right Kleene algebra (RKA)* if $(S, \Sigma, \breve{\cdot}, 0, 1)$ is a *left Kleene algebra*. A *Kleene algebra (KA)* (cf. [4]) is a quintuple $(S, \Sigma, \cdot, 0, 1)$ which is both an LKA and an RKA.

In connection with graph algorithms, one often considers the related structure of a *closed semiring* (see e.g. [1]). It differs from a KA in that ΣK is only required to exist for *countable* K; moreover, idempotence of $+$ is not postulated. So every KA is a closed semiring, but not vice versa.

Perhaps the best-known example of a KA is

$$\text{LAN} \stackrel{\text{def}}{=} (\mathcal{P}(A^*), \ \bigcup, \ \bullet, \ \emptyset, \ \varepsilon),$$

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). Here $\leq$ coincides with $\subseteq$.

3 Typed Kleene Algebras

3.1 Definition and Basic Properties

A *subidentity* of an LKA is an element x with $x \leq 1$. We call an LKA *pre-typed* if all its subidentities are idempotent, ie. if $x \leq 1 \ \Rightarrow \ x \cdot x = x$.

In a pre-typed LKA/RKA the subidentities play the role of types. This is best illustrated with the KA of binary relations over some universe U under set union as the interpretation of Σ, relational composition as the interpretation of $\cdot$ and the identity relation as the interpretation of 1. A type corresponds to a subset $T \subseteq U$ of the universe and can be represented as the partial identity relation $1_T \stackrel{\text{def}}{=} \{(x,x) \mid x \in T\}$. Clearly, 1_T is a subidentity, and so there is a one-to-one correspondence between types and subidentities.

Now restriction of a relation $R \subseteq U \times U$ to arguments of type T, ie. the relation $R \cap T \times U$, can also be described by composing R with 1_T from the left:

$$R \cap T \times U = 1_T \cdot R \; .$$

Similarly, co-restriction is modeled by composing a partial identity from the right. Finally, consider types $S, T \subseteq U$ and binary relation $R \subseteq U \times U$. Then

$$R \subseteq S \times T \;\Leftrightarrow\; 1_S \cdot R \cdot 1_T = R \; .$$

In other words, the "default typing" $U \times U$ of R can be narrowed down to $S \times T$ iff restriction to S and co-restriction to T does not change R.

These observations are the basis for our view of (pre-)types as subidentities and our algebraic treatment of restriction and co-restriction. We have

Lemma 31 *In a pre-typed LKA the infimum of two types is their product:*

$$x, y \leq 1 \Rightarrow x \cdot y = x \sqcap y \; .$$

In particular, all types commute under the $\cdot$ operation.

Proof: First, by monotonicity of $\cdot$ it is clear that $x \cdot y \leq x, y$. Assume now $z \leq x, y$. By transitivity, $z \leq 1$ and hence

$$\begin{array}{ll} & z \\ = & \quad \{\!\![\text{ idempotence }]\!\!\} \\ & z \cdot z \\ \leq & \quad \{\!\![\text{ monotonicity }]\!\!\} \\ & x \cdot y \; . \end{array}$$

■

We call a pre-typed LKA *typed* if it is a boolean algebra and the restriction operation distributes through arbitrary meets of subtypes, i.e., if we have for all sets K of subidentities and all $a \in S$ that

$$(\sqcap K) \cdot a = \sqcap (K \cdot a) \; , \tag{3}$$

where on the right hand side $\cdot$ means the pointwise extension of the original $\cdot$ operation to sets. Then the subidentities are called *types*.

An RKA is called *pre-typed* and *typed*, resp., if its associated LKA is.

In a typed LKA or RKA we denote by

$$\text{TYP} \stackrel{\text{def}}{=} \{x \in M : x \leq 1\}$$

the set of types.

Lemma 32 *Consider an LKA. Then, for* $x, y \in \mathrm{TYP}$*,*

$$x \cdot y \cdot \top \;=\; x \cdot \top \sqcap y \cdot \top \;.$$

Proof:

$$\begin{array}{ll} & x \cdot \top \sqcap y \cdot \top \\ = & \quad \{\![\text{ typedness }]\!\} \\ & (x \sqcap y) \cdot \top \\ = & \quad \{\![\text{ by Lemma 31 }]\!\} \\ & x \cdot y \cdot \top \;. \end{array}$$

■

3.2 Domain and Codomain

In a typed LKA we can define, for $a \in S$, the *domain* $\lceil a$ via the Galois connection

$$\forall y \leq 1 : \lceil a \leq y \;\stackrel{\text{def}}{\Leftrightarrow}\; a \leq y \cdot \top \;.$$

This is well-defined because of assumption (3). Hence the operation $\lceil\_$ is universally disjunctive and therefore monotonic and strict. Moreover the definition implies

$$a \leq \lceil a \cdot \top \;. \tag{4}$$

By the Galois connection the partial orders $(\mathrm{TYP}, \leq)$ and $(\{x \cdot \top \mid x \in \mathrm{TYP}\}, \leq)$ are isomorphic. Hence we have, for $x \in \mathrm{TYP}$,

$$\lceil(x \cdot \top) \;=\; x$$

(which also follows from properties 7, 4 and 3 in Lemma 33 below).

For defining the *co-domain* $a\rceil$ one uses a typed RKA and the Galois connection

$$a\rceil \leq y \;\stackrel{\text{def}}{\Leftrightarrow}\; a \leq \top \cdot y \;.$$

We can now show the usual properties of the domain operation; the analogous ones for the co-domain operation follow by passing to the LKA associated with a given RKA.

Lemma 33 *Consider a typed LKA and* $a, b, c \in S$*.*

1. $\lceil a \;=\; \min\{x : x \leq 1 \wedge x \cdot a = a\}$*. In particular,*
   (a) $\lceil a \cdot a \;=\; a$ *,*
   (b) $x \leq 1 \wedge x \cdot a = a \;\Rightarrow\; \lceil a \leq x$ *.*
2. $\lceil(a \cdot b) \leq \lceil a$ *.*
3. $x \leq 1 \;\;\Rightarrow\;\; \lceil x \;=\; x$ *.*
4. $\lceil \top \;=\; 1$ *.*
5. $\lceil(\lceil a) \;=\; \lceil a$ *.*
6. $a \cdot \top \leq \lceil a \cdot \top$ *.*
7. $\lceil(a \cdot b) \leq \lceil(a \cdot \lceil b)$ *.*
8. $\lceil a \;=\; 0 \;\Leftrightarrow\; a \;=\; 0$ *.*

Proof: 1. Let $D \stackrel{\text{def}}{=} \{x : x \leq 1 \wedge x \cdot a = a\}$ and $z \stackrel{\text{def}}{=} \sqcap D$. Note that D is closed under the $\cdot$ operation. We have

$$
\begin{array}{ll}
 & z \cdot a \\
= & \quad \{\![\text{ definition } z \ ]\!\} \\
 & (\sqcap D) \cdot a \\
= & \quad \{\![\text{ typedness }]\!\} \\
 & \sqcap (D \cdot a) \\
= & \quad \{\![\text{ definition } D \ ]\!\} \\
 & \sqcap \{a\} \\
= & \quad \{\![\text{ infimum }]\!\} \\
 & a
\end{array}
$$

and hence $z \in D$. Therefore, even $z = \min D$. Now by $a \leq \top$ and monotonicity we get $a = z \cdot a \leq z \cdot \top$ and hence, by the Galois connection $\ulcorner a \leq z$.

Again from the Galois connection we get $a \leq \ulcorner a \cdot \top$. Hence

$$
\begin{array}{ll}
 & a \\
= & \quad \{\![\text{ order in lattices }]\!\} \\
 & a \sqcap \ulcorner a \cdot \top \\
= & \quad \{\![\text{ boolean algebra and disjunctivity }]\!\} \\
 & a \sqcap (\ulcorner a \cdot a + \ulcorner a \cdot \overline{a}) \\
= & \quad \{\![\text{ boolean algebra }]\!\} \\
 & (a \sqcap \ulcorner a \cdot a) + (a \sqcap \ulcorner a \cdot \overline{a}) \\
= & \quad \{\![\text{ since, by } \ulcorner a \leq 1, \text{ we have } a \sqcap \ulcorner a \cdot \overline{a} \leq a \sqcap \overline{a} = 0 \ ]\!\} \\
 & a \sqcap \ulcorner a \cdot a \ ,
\end{array}
$$

so that $a \leq \ulcorner a \cdot a$. Since $\ulcorner a \cdot a \leq a$ is trivial, we get $\ulcorner a \in D$ and hence $z \leq \ulcorner a$.

2. We have

$$
\begin{array}{ll}
 & \ulcorner a \cdot (a \cdot b) \\
= & \quad \{\![\text{ associativity }]\!\} \\
 & (\ulcorner a \cdot a) \cdot b \\
= & \quad \{\![\text{ by 1 }]\!\} \\
 & a \cdot b \ ,
\end{array}
$$

so that the claim follows by 1.

3. By pre-typedness, $x = x \cdot x$, and hence $\ulcorner x \leq x$ by 1. Now

$$
\begin{array}{ll}
 & \ulcorner x \\
= & \quad \{\![\text{ order in lattices }]\!\} \\
 & \ulcorner x \sqcap x \\
= & \quad \{\![\text{ Lemma 31 }]\!\} \\
 & \ulcorner x \cdot x \\
= & \quad \{\![\text{ by 1. }]\!\} \\
 & x \ .
\end{array}
$$

4. The inequation $\ulcorner \top \leq 1$ holds by definition of $\ulcorner \_$. The reverse inequation follows from $1 \leq \top$ and monotonicity of $\ulcorner \_$.
5. is immediate from 3.

6.
$$\begin{aligned} & a \cdot \top \\ \leq & \quad \{\![\text{ by (4) }]\!\} \\ & \ulcorner a \cdot \top \cdot \top \\ = & \quad \{\![\text{ by Corollary 21 }]\!\} \\ & \ulcorner a \cdot \top \ . \end{aligned}$$

7.
$$\begin{aligned} & \ulcorner(a \cdot b) \cdot a \cdot b \\ = & \quad \{\![\text{ by 1 }]\!\} \\ & \ulcorner(a \cdot b) \cdot a \cdot \ulcorner b \cdot b \\ = & \quad \{\![\text{ by 1 }]\!\} \\ & a \cdot \ulcorner b \cdot b \\ = & \quad \{\![\text{ by }]\!\} \\ & a \cdot b \ . \end{aligned}$$

Now the claim follows by 1.

8.
$$\begin{aligned} & \ulcorner a \leq 0 \\ \Leftrightarrow & \quad \{\![\text{ by the defining Galois connection }]\!\} \\ & a \leq 0 \cdot \top \\ \Leftrightarrow & \quad \{\![\text{ by 0-strictness of } \cdot \ ]\!\} \\ & a \leq 0 \ . \end{aligned}$$

■

According to Lemma 33. 8 the domain of an element also decides its "definedness" if we identify 0 with $\bot$ as used in denotational semantics.

It should be noted that the converse inequation to Lemma 33. 7 does not follow from our axiomatisation. A counterexample has been found by J. Desharnais. 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 an LKA has *local composition* if it satisfies

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

To check this property, by left-strictness of $\cdot$ it suffices to consider $a \neq 0$. If $\cdot$ is right-strict as well, by Lemma 33.2 one need only consider b, c with $\ulcorner b = \ulcorner c \neq 0$.

Lemma 34 *1. An LKA has local composition iff it satisfies*

$$\ulcorner(a \cdot b) = \ulcorner(a \cdot \ulcorner b) \tag{5}$$

2. If an LKA has local composition then

$$\ulcorner(\ulcorner a \cdot b) = \ulcorner a \sqcap \ulcorner b = \ulcorner a \cdot \ulcorner b \ .$$

3. If an LKA has local composition then

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

Proof: 1. Assume local composition. Then (5) follows, since $\ulcorner b = \ulcorner(\ulcorner b)$ by Lemma 33. 5. Now assume (5) and $\ulcorner b = \ulcorner c$. Then

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

2. Immediate from (5), Lemma 33. 3 and Lemma 31.
3. Immediate from (5) and monotonicity.

■

Another useful property is

Lemma 35 *If the element* 1 *of an LKA is also right-neutral w.r.t.* $\cdot$ *then*

$$\ulcorner(a \cdot \top) = \ulcorner a \ .$$

Proof: The inequation $\leq$ follows from Lemma 33.2, whereas $\geq$ follows from $a \cdot \top \geq a \cdot 1 = a$ and monotonicity of $\ulcorner\_$. ■

Finally we note

Lemma 36 *In a typed KA we have*

$$a^{\urcorner} \sqcap \ulcorner b = 0 \ \Rightarrow \ a \cdot b = 0 \ .$$

Proof:

$$\begin{aligned}
& a \cdot b \\
= \quad & \{\![\text{ by Lemma 33.1 }]\!\} \\
& a \cdot a^{\urcorner} \cdot \ulcorner b \cdot b \\
= \quad & \{\![\text{ by Lemma 31 }]\!\} \\
& a \cdot (a^{\urcorner} \sqcap \ulcorner b) \cdot b \\
= \quad & \{\![\text{ assumption }]\!\} \\
& a \cdot 0 \cdot b \\
= \quad & \{\![\text{ strictness of } \cdot \ ]\!\} \\
& 0 \ .
\end{aligned}$$

■

4 Particular Typed Kleene Algebras

4.1 Formal Languages and Relations

We have already introduced the Kleene algebra LAN of formal languages over an alphabet A. In connection with graph algorithms the letters of A can be interpreted as nodes and the words of a language can be used to represent paths in that graph: the nodes are listed in the order of traversal.

A *relation* is a language R in which all words have equal length. This length is called the *arity* of the relation, in symbols $\mathsf{ar}\, R$. The empty relation $\emptyset$ has all arities. Unary relations can be interpreted as sets of nodes, whereas binary relations represent sets of edges. The binary *identity relation* is

$$I \stackrel{\text{def}}{=} \{a \bullet a : a \in A\} \ ,$$

whereas $A \bullet A$ is the *universal relation* on A.

4.2 Join and Composition

For words s and t over alphabet A we now define their **join** $s \bowtie t$ and their **composition** $s\,;t$ as set-valued operations. Remember that we identify singleton languages with their only elements. With this convention we set

$$\varepsilon \bowtie \varepsilon \stackrel{\text{def}}{=} \varepsilon \ , \qquad \varepsilon\,;\varepsilon \stackrel{\text{def}}{=} \varepsilon \ ,$$

as well as, for $s \in A^+$,

$$\varepsilon \bowtie s \stackrel{\text{def}}{=} \emptyset \stackrel{\text{def}}{=} s \bowtie \varepsilon \ , \qquad \varepsilon\,;s \stackrel{\text{def}}{=} \emptyset \stackrel{\text{def}}{=} s\,;\varepsilon \ ,$$

and, for $s,t \in A^*$ and $x,y \in A$,

$$\begin{aligned} (s \bullet x) \bowtie (y \bullet t) &\stackrel{\text{def}}{=} \begin{cases} s \bullet x \bullet t \text{ if } x = y \ , \\ \emptyset \qquad\quad \text{otherwise} \ , \end{cases} \\ (s \bullet x)\ ;\ (y \bullet t) &\stackrel{\text{def}}{=} \begin{cases} s \bullet t \text{ if } x = y \ , \\ \emptyset \quad \text{otherwise} \ . \end{cases} \end{aligned}$$

These operations provide two different ways of "gluing" two words together upon a one-letter overlap: join preserves one copy of the overlap, whereas composition erases it. They are extended pointwise to languages; hence they are universally disjunctive (see eg. [9] for details).

On binary relations, composition coincides with usual relational composition (see e.g. [15]). To save parentheses we use the convention that $\bullet$, $\bowtie$ and ; bind stronger than all set-theoretic operations.

To exemplify the close connection between join and composition further, we consider a binary relation $R \subseteq A \bullet A$ modeling the edges of a directed graph with node set A. Then the relation

$$R \bowtie R \ = \ \{x \bullet z \bullet y \ : \ x \bullet z \in R \wedge z \bullet y \in R\}$$

consists of exactly those paths $x \bullet z \bullet y$ which result from gluing two edges together at a common intermediate node. The composition $R\,;R$ is an abstraction of this; it just states whether there is a path from x to y via some intermediate point without making that point explicit.

Iterating this observation shows that the relations

$$R,\ R \bowtie R,\ R \bowtie (R \bowtie R),\ \ldots$$

consist of the paths with exactly $1,2,3,\ldots$ edges in the directed graph associated with R, whereas the relations

$$R,\ R\,;R,\ R\,;(R\,;R),\ \ldots$$

just state existence of these paths between pairs of vertices.

4.3 Further Examples of KAs

Now we can give three further examples of KAs:

$$\begin{aligned} \text{REL} &\stackrel{\text{def}}{=} (\mathcal{P}(A \bullet A),\ \bigcup,\ ;,\ \emptyset,\ I), \\ \text{PAT} &\stackrel{\text{def}}{=} (\mathcal{P}(A^*),\ \bigcup,\ \bowtie,\ \emptyset,\ A \cup \varepsilon) \ , \\ \text{NEPAT} &\stackrel{\text{def}}{=} (\mathcal{P}(A^+),\ \bigcup,\ \bowtie,\ \emptyset,\ A) \ . \end{aligned}$$

More generally than the concrete relation algebra REL, a subalgebra of every abstract relation algebra is a KA. Such an *abstract relation algebra* (see eg. [15] is a tuple

$$\mathrm{RA} \;=\; (M, \bigsqcup, \bar{\_}, \top, ;, 0, 1, \breve{\_}) \;,$$

where

1. $(M, \bigsqcup, \bar{\_}, 0, \top)$ is a Boolean algebra, i.e., a complete, distributive, atomic, and complementary lattice with supremum operation $\bigsqcup$, complement operation $\bar{\_}$, least element 0 and greatest element $\top$. The corresponding lattice order is denoted by $\leq$; the binary supremum operator $+$ and infimum operator $\sqcap$ are given by
$$x + y \stackrel{\text{def}}{=} \bigsqcup \{x, y\} \;,$$
$$x \sqcap y \stackrel{\text{def}}{=} \overline{\overline{x} + \overline{y}} \;.$$
2. $(M, ;, 1)$ is a monoid.
3. Tarski's rule $x \neq 0 \Rightarrow \top ; x ; \top = \top$ holds.
4. Dedekind's rule $x ; y \sqcap z \;\leq\; (x \sqcap z ; y^{\smile}) ; (y \sqcap x^{\smile}; z)$ is satisfied.

The elements of M are called *abstract relations*; the operation $\breve{\_}$ forms the *converse* of a relation, whereas ; is called *relational composition*. It is customary to use the convention that ; binds tighter than $+$ and $\sqcap$. Now the tuple

$$(M, \bigsqcup, ;, 0, 1)$$

forms a KA which we again denote by RA.

A good deal of the remainder of the paper is devoted to the proof of the following

Theorem 41 *The KAs* LAN, RA, PAT *and* NEPAT *are all typed. Moreoever, we have the following explicit representations of the domain operation.*

1. *In* LAN*:* $\ulcorner U = \delta U$ *where*
$$\delta U \stackrel{\text{def}}{=} \begin{cases} \varepsilon \text{ if } U \neq \emptyset, \\ \emptyset \text{ otherwise.} \end{cases}$$
2. *In* RA*:* $\ulcorner x = x ; \top \sqcap 1$.
3. *In* PAT *and* NEPAT*:* $\ulcorner P = \mathit{first}(P)$ *where*
$$\mathit{first}(P) \stackrel{\text{def}}{=} \{x \in A : x \bullet A^* \cap P \neq \emptyset\} \cup (P \cap \varepsilon) \;.$$

Here, as is customary in formal language theory, a singleton language is identified with its only word to save braces; moreover, a word consisting just of one letter is not distinguished from that letter.

We start the proof with

Lemma 42 *The KAs* LAN, RA, PAT *and* NEPAT *are pre-typed.*

Proof: First we note that, for $U, V \subseteq A \cup \varepsilon$,

$$U \bowtie V \;=\; U \cap V \;, \qquad\qquad U ; V \;=\; \delta(U \cap V) \;.$$

Moreover,

$$I_B ; I_C \;=\; I_{B \cap C} \;.$$

From this, pre-typedness of LAN, REL, PAT and NEPAT is straightforward. For pre-typedness of RA we note that

$$
\begin{array}{ll}
 & x \\
= & \quad \{\![\text{ neutrality }]\!\} \\
 & x \,;\, 1 \sqcap \top \\
\leq & \quad \{\![\text{ Dedekind }]\!\} \\
 & (x \sqcap \top \,;\, 1^{\smile}) \,;\, (1 \sqcap x^{\smile} ; \top) \\
\leq & \quad \{\![\text{ infimum }]\!\} \\
 & x \,;\, (1 \sqcap x^{\smile} ; \top) \\
\leq & \quad \{\![\text{ Dedekind }]\!\} \\
 & x \,;\, (x^{\smile} \sqcap 1 \,;\, \top^{\smile}) \,;\, (\top \sqcap (x^{\smile})^{\smile} ; 1) \\
\leq & \quad \{\![\text{ neutrality, infimum, } (x^{\smile})^{\smile} = x \,]\!\} \\
 & x \,;\, x^{\smile} ; x
\end{array}
$$

for arbitrary $x \in M$. If even $x \leq 1$ (and hence also $x^{\smile} \leq 1$) this implies $x \leq x \,;\, x$. The converse inequation $x \,;\, x \leq x$ for $x \leq 1$ is immediate by monotonicity of the ; operation. ■

4.4 Proofs for LAN

Next we show typedness of LAN, ie. distributivity of restriction by a type over arbitrary intersections. Let $(B_i)_{i \in I}$ be a family of types, ie. $B_i \subseteq \varepsilon$ for all $i \in I$, and consider $U \subseteq A^*$.

Case 1: $\bigcap_{i \in I} B_i = \emptyset$. Then there is an $i_0 \in I$ with $B_{i_0} = \emptyset$. Then also $B_{i_0} \bullet U = \emptyset$, so that $\bigcap_{i \in I} (B_i \bullet U) = \emptyset = (\bigcap_{i \in I} B_i) \bullet U$.

Case 2: $\bigcap_{i \in I} B_i = \varepsilon$. Then $B_i = \varepsilon$ for all $i \in I$ and hence $\bigcap_{i \in I} (B_i \bullet U) = U = (\bigcap_{i \in I} B_i) \bullet U$.

To verify the expression for the domain operator, we just need to check that the defining Galois equation holds for it.

1. **Case 1:** $U = \emptyset$. Then
$$\delta U \subseteq Y \Leftrightarrow U \subseteq Y \bullet \top$$
is trivial.
Case 2: $U \neq \emptyset$. Then
$$
\begin{array}{ll}
 & \delta U \subseteq Y \\
\Leftrightarrow & \quad \{\![\text{ definition of } \delta \,]\!\} \\
 & \varepsilon \subseteq Y \\
\Rightarrow & \quad \{\![\text{ since } \varepsilon \in \top \,]\!\} \\
 & \varepsilon \subseteq Y \bullet \top \,.
\end{array}
$$
Conversely,
$$
\begin{array}{ll}
 & \varepsilon \subseteq Y \bullet \top \\
\Rightarrow & \quad \{\![\text{ strictness }]\!\}
\end{array}
$$

$$
\begin{array}{ll}
& \varepsilon \subseteq Y \bullet \top \wedge Y \neq \emptyset \\
\Rightarrow & \quad \{\![\text{ indivisibility of } \varepsilon \ ]\!\} \\
& \varepsilon \subseteq Y \\
\Leftrightarrow & \quad \{\![\text{ definition of } \delta \ ]\!\} \\
& \delta U \subseteq Y \ .
\end{array}
$$

We conclude this section with an additional domain law that holds in LAN but not in our other examples of KAs (the proof is straightforward):

Lemma 43 *In* LAN*,*

$$\ulcorner(U \cdot V) = \ulcorner U \cdot \ulcorner V \ .$$

4.5 Proofs for PAT

Next we consider PAT (which includes NEPAT). Let $(B_i)_{i \in I}$ be a family of types, ie. $B_i \subseteq A \cup \varepsilon$ for all $i \in I$, and consider $U \subseteq A^*$. We perform again a case analysis. First,

$$
\begin{array}{ll}
& \varepsilon \in (\bigcap_{i \in I} B_i) \bowtie U \\
\Leftrightarrow & \quad \{\![\text{ definition of } \bowtie \ ]\!\} \\
& \varepsilon \in (\bigcap_{i \in I} B_i) \cap U \\
\Leftrightarrow & \quad \{\![\text{ set theory }]\!\} \\
& \varepsilon \in \bigcap_{i \in I} (B_i \cap U) \\
\Leftrightarrow & \quad \{\![\text{ definition of } \bowtie \ ]\!\} \\
& \varepsilon \in \bigcap_{i \in I} (B_i \bowtie U) \ .
\end{array}
$$

Second, consider $s \in A^+$.

$$
\begin{array}{ll}
& s \in (\bigcap_{i \in I} B_i) \bowtie U \\
\Leftrightarrow & \quad \{\![\text{ definition of } \bowtie \ ]\!\} \\
& \mathit{first}(s) \subseteq \bigcap_{i \in I} B_i \ \wedge \ s \in U \\
\Leftrightarrow & \quad \{\![\text{ definition of intersection }]\!\} \\
& (\forall \ i \in I : \mathit{first}(s) \subseteq B_i) \wedge s \in U \\
\Leftrightarrow & \quad \{\![\text{ logic }]\!\} \\
& \forall \ i \in I : (\mathit{first}(s) \subseteq B_i \wedge s \in U) \\
\Leftrightarrow & \quad \{\![\text{ definition of } \bowtie \ ]\!\} \\
& \forall \ i \in I : s \in B_i \bowtie U \\
\Leftrightarrow & \quad \{\![\text{ set theory }]\!\} \\
& s \in \bigcap_{i \in I} (B_i \bowtie U) \ .
\end{array}
$$

The Galois connection for *first* is clear from the definition of $\bowtie$.

This time we mention a domain law that holds for PAT but not for our other examples of KAs. Again, the proof is straightforward.

Lemma 44 *For $u, v \in A^*$ such that $u \bowtie v \neq \emptyset$ we have*

$$\ulcorner u \bowtie v = \ulcorner u \ .$$

4.6 Proofs for RA

Here we deviate from the general scheme of our proofs. First, we show a number of auxiliary properties. After that, we prove — without using typedness — that the claimed expression for the domain satisfies the required Galois connection. This will finally be used to establish typedness.

We start with the following

Lemma 45 *For $x \leq 1$ and arbitrary $a \in M$ we have*

$$x \,;\, a = a \sqcap x \,;\, \top \ .$$

Proof: First, by $x \leq 1$ and monotonicity we get $x \,;\, a \leq a$. Second, by $a \leq \top$ and monotonicity we get $x \,;\, a \leq x \,;\, \top$. Hence

$$x \,;\, a \leq a \sqcap x \,;\, \top \ .$$

For the converse inequation we calculate

$$\begin{array}{ll} & x \,;\, \top \sqcap a \\ \leq & \quad \{\![\text{ Dedekind }]\!\} \\ & (x \sqcap a \,;\, \top^\smile) \,;\, (\top \sqcap x^\smile ; a) \\ \leq & \quad \{\![\text{ by monotonicity and } x^\smile = x \text{ for } x \leq 1 \ ]\!\} \\ & x \,;\, x \,;\, a \\ = & \quad \{\![\text{ pre-typedness }]\!\} \\ & x \,;\, a \ . \end{array}$$

■

Define now

$$\partial a \stackrel{\text{def}}{=} 1 \sqcap a \,;\, \top \ .$$

Corollary 46 *For $x \leq 1$ we have $x = \partial x$.*

Proof: Choose $a = 1$ in Lemma 45. ■

Moreover,

Lemma 47 *1. $a \leq \partial a \,;\, \top$.*
2. $\partial a \,;\, a = a$.

Proof: 1.

$$\begin{array}{ll} & a \\ = & \quad \{\![\text{ relational algebra }]\!\} \\ & 1 \,;\, \top \sqcap a \\ \leq & \quad \{\![\text{ Dedekind }]\!\} \\ & (1 \sqcap a \,;\, \top^\smile) \,;\, (\top \sqcap 1^\smile ; a) \end{array}$$

$$
\begin{array}{ll}
\leq & \{\![\text{ monotonicity }]\!\} \\
& (1 \sqcap a \,;\, \top^{\smile}) \,;\, \top \\
\leq & \{\![\text{ relational algebra }]\!\} \\
& (1 \sqcap a \,;\, \top) \,;\, \top \,.
\end{array}
$$

2. By Lemma 45 we have

$$\partial a \,;\, a \;=\; a \sqcap \partial a \,;\, \top \,.$$

Now apply 1.

■

Now we are ready to show

Lemma 48 *For $y \leq 1$ and arbitrary $a \in M$,*

$$\partial a \leq y \;\Leftrightarrow\; a \leq y \,;\, \top \,.$$

Proof: ($\Rightarrow$)

$$
\begin{array}{ll}
& \partial a \leq y \\
\Rightarrow & \{\![\text{ monotonicity }]\!\} \\
& \partial a \,;\, \top \leq y \,;\, \top \\
\Rightarrow & \{\![\text{ by } a \leq \top \text{ and monotonicity }]\!\} \\
& \partial a \,;\, a \leq y \,;\, \top \\
\Leftrightarrow & \{\![\text{ by Lemma 47.2 }]\!\} \\
& a \leq y \,;\, \top \,.
\end{array}
$$

($\Leftarrow$)

$$
\begin{array}{ll}
& a \leq y \,;\, \top \\
\Rightarrow & \{\![\text{ monotonicity }]\!\} \\
& a \,;\, \top \leq y \,;\, \top \,;\, \top \\
\Leftrightarrow & \{\![\text{ relational algebra }]\!\} \\
& a \,;\, \top \leq y \,;\, \top \\
\Rightarrow & \{\![\text{ monotonicity }]\!\} \\
& a \,;\, \top \sqcap 1 \leq y \,;\, \top \sqcap 1 \\
\Leftrightarrow & \{\![\text{ by Corollary 46 }]\!\} \\
& a \,;\, \top \sqcap 1 \leq y \\
\Leftrightarrow & \{\![\text{ definition of } \partial \,]\!\} \\
& \partial a \leq y \,.
\end{array}
$$

■

This lemma means that ∂ and f with $f(x) \stackrel{\text{def}}{=} x \,;\, \top$ form a Galois connection between $(M, \leq)$ and $(\text{TYP}, \leq)$. From this we obtain

Corollary 49 RA *is typed.*

Proof: As the upper adjoint of a Galois connection, f distributes through arbitrary infima. Consider now a subset $K \subseteq \text{TYP}$ and an element $a \in M$. We calculate

$$
\begin{array}{ll}
 & a \\
= & \quad \{\![\text{ by Lemma 45 }]\!\} \\
 & a \sqcap (\sqcap K)\,;\top \\
= & \quad \{\![\text{ distributivity of } f \text{ over } \sqcap \ ]\!\} \\
 & a \sqcap (\sqcap (K\,;\top)) \\
= & \quad \{\![\text{ distributivity of } \sqcap \text{ over } \sqcap \ ]\!\} \\
 & \sqcap (a \sqcap K\,;\top) \\
= & \quad \{\![\text{ by Lemma 45 }]\!\} \\
 & \sqcap (a\,;K)\ .
\end{array}
$$

■

Finally, since the lower adjoint of a Galois connection is uniquely determined by the upper one, we have $\partial a = \ulcorner a$.

4.7 Locality of Composition

We conclude this chapter by stating

Lemma 410 *The LKAs* LAN, PAT *and* RA *have local composition and hence satisfy (5).*

Proof: For LAN this is immediate from the definitions.

For PAT we note that

$$\ulcorner(U \bowtie V) = \{\mathit{first}(u) : u \in U \wedge \mathit{last}(u) \cap \ulcorner V \neq \emptyset\}\ .$$

Consider finally RA. First we note that

$$
\begin{array}{ll}
 & \ulcorner a \cdot \top \\
= & \quad \{\![\text{ by Theorem 41 }]\!\} \\
 & (1 \sqcap a \cdot \top) \cdot \top \\
\leq & \quad \{\![\text{ infimum and monotonicity }]\!\} \\
 & a \cdot \top \cdot \top \\
= & \quad \{\![\text{ by Corollary 21 }]\!\} \\
 & a \cdot \top\ ,
\end{array}
$$

so that by Lemma 33.6 we have

$$\ulcorner a \cdot \top = a \cdot \top\ . \tag{6}$$

Now

$$
\begin{array}{ll}
 & \ulcorner(a \cdot b) \\
= & \quad \{\![\text{ by Lemma 35 }]\!\} \\
 & \ulcorner(a \cdot b \cdot \top) \\
= & \quad \{\![\text{ by (6) }]\!\} \\
 & \ulcorner(a \cdot \ulcorner b \cdot \top) \\
= & \quad \{\![\text{ by Lemma 35 }]\!\} \\
 & \ulcorner(a \cdot \ulcorner b)\ .
\end{array}
$$

■

The proof shows that equation (6) implies locality of composition. However, it is not equivalent to that postulate. As a counterexample, consider the KA PAT which has local composition but does not satisfy (6).

5 Truth Values and Assertions

The elements 1 and 0 of a Kleene algebra can play the roles of the truth values "true" and "false". Expressions that yield one of these values are therefore also called *assertions* (see e.g. [11]). The assertion 0 means not only "false", but also "undefined".

Negation is defined by

$$\neg 0 \stackrel{\text{def}}{=} 1 \ , \qquad \neg 1 \stackrel{\text{def}}{=} 0 \ .$$

Note that negation is not $\leq$-monotonic.

For an assertion b and an element c we have

$$b \cdot c = c \cdot b = \begin{cases} c \text{ if } b = 1 \ , \\ 0 \text{ if } b = 0 \ . \end{cases}$$

Note that 0 and 1 are types. The conjunction of types and hence assertions a, b is their infimum $a \sqcap b$ or, equivalently, their product $a \cdot b$; their disjunction is their sum $a + b$. We write $a \wedge b$ for $a \sqcap b$ and $a \vee b$ for $a + b$.

Using assertions we can construct a conditional and a guarded expression:

$$\begin{aligned} &\mathsf{if}\ b\ \mathsf{then}\ c\ \mathsf{else}\ d\ \mathsf{fi} \stackrel{\text{def}}{=} b \cdot c \ + \ \neg b \cdot d \ , \\ &\mathsf{if}\ b_1\ \mathsf{then}\ c_1\ [\!]\ \cdots\ [\!]\ b_n\ \mathsf{then}\ c_n\ \mathsf{fi} \stackrel{\text{def}}{=} \sum_{i=1}^{n} b_i \cdot c_i \ . \end{aligned}$$

for assertions b, b_i and elements c, c_i, d. Note that the conditional is monotonic only in c and d. So recursions over the condition b need not be well-defined.

As an example for working with assertions we show

$$\mathsf{if}\ b\ \mathsf{then}\ d\ \mathsf{else}\ \mathsf{if}\ c\ \mathsf{then}\ d\ \mathsf{else}\ e\ \mathsf{fi}\ \mathsf{fi} = \mathsf{if}\ b \vee c\ \mathsf{then}\ d\ \mathsf{else}\ e\ \mathsf{fi} \tag{7}$$

for assertions b, c and elements d, e:

$$\begin{aligned} & \mathsf{if}\ b\ \mathsf{then}\ d\ \mathsf{else}\ \mathsf{if}\ c\ \mathsf{then}\ d\ \mathsf{else}\ e\ \mathsf{fi}\ \mathsf{fi} \\ = \quad & \{\![\text{ definition of } \mathsf{if} \ ]\!\} \\ & b \cdot d \ + \ \neg b \cdot (c \cdot d \ + \ \neg c \cdot e) \\ = \quad & \{\![\text{ distributivity }]\!\} \\ & b \cdot d \ + \ \neg b \cdot c \cdot d \ + \ \neg b \cdot \neg c \cdot e \\ = \quad & \{\![\text{ distributivity }]\!\} \\ & (b \ + \ \neg b \cdot c) \cdot d \ + \ \neg b \cdot \neg c \cdot e \\ = \quad & \{\![\text{ typedness }]\!\} \\ & (b \vee (\neg b \wedge c)) \cdot d \ + \ (\neg b \wedge \neg c) \cdot e \\ = \quad & \{\![\text{ Boolean algebra }]\!\} \end{aligned}$$

$$
\begin{array}{ll}
 & (b \vee c) \cdot d \ + \ \neg(b \vee c) \cdot e \\
= & \quad \{\![\text{ definition of } \mathsf{if} \ ]\!\} \\
 & \mathsf{if} \ b \vee c \, \mathsf{then} \, d \, \mathsf{else} \, e \ \mathsf{fi} \ .
\end{array}
$$

6 Application: Efficiency Increase of a Schematic Algorithm

In [3] we have presented an abstraction of layer-oriented graph traversal in general KAs. The original form of the algorithm was transformed into a more efficient one using an particular invariant.

In this section we sketch the derivation and fill in the proof of preservation of the invariant.

We first repeat the definition of the function F for which we want to develop an efficient algorithm. It reads

$$
F(f,g)(a,b,c) \stackrel{\text{def}}{=} E(f,g)(a \cdot b^* \cdot c) = g(f(a \cdot b^* \cdot c)) \ ,
$$

with $a, b, c \in S$. Here, a, b, c are elements of a KA $(S, \Sigma, \cdot, 0, 1)$ and f and g satisfy the following requirements:

- $w \in S$ is a fixed element of S,
- $f : S \to \mathcal{P}(M)$ is a disjunctive *abstraction* function with some set M of "valuations", where a function f from a Kleene algebra into an upper semilattice is *disjunctive* if it distributes through $+$, i.e., satisfies $f(x + y) = f(x) \sqcup f(y)$,
- $g : \mathcal{P}(M) \to \mathcal{P}(M)$ is a *selection* satisfying the properties

  $$
  \begin{array}{ll}
  \text{(GEN1)} & g(K) \subseteq K \ , \\
  \text{(GEN2)} & g(K \cup L) = g(g(K) \cup g(L)) \text{ (weak distributivity)},
  \end{array}
  $$

  for $K, L \subseteq M$.
- f and g jointly satisfy the laws

  $$
  \begin{array}{ll}
  \text{(LAY1)} & \ulcorner c \ \leq \ a \urcorner \ \Rightarrow \ g(f(a \cdot c) \cup f(a \cdot u \cdot c)) = g(f(a \cdot c)) \ , \\
  \text{(LAY2)} & (a \cdot u)\urcorner \ \leq \ a \urcorner \ \Rightarrow \ g(f(a \cdot c) \cup f(a \cdot u \cdot c)) = g(f(a \cdot c)) \ ,
  \end{array}
  $$

  with $a, c, u \in S$.

Lt us interpret this in the particular Kleene algebra PAT. The expression $a \cdot b^* \cdot c$ denotes the set of all paths that connect a and c and exclusively use pieces from b. In (LAY1), $\ulcorner c \leq a \urcorner$ is the case where the set of starting nodes of c already is contained in the set of end nodes of a. The condition $(a \cdot u)\urcorner \leq a\urcorner$ in (LAY2) means that by further traversal of the graph along u no new end nodes are reached. In both cases the second use of the abstraction f should give no new information and be ignored by g.

From the definition one can derive the following basic algorithm:

$$
\begin{array}{ll}
F(f,g)(a,b,c) = \mathsf{if} \ \ulcorner c \ \leq \ a \urcorner \ \vee \ (a \cdot b)\urcorner \ \leq \ a\urcorner & \\
\qquad \mathsf{then} \, g(f(a \cdot c)) & \\
\qquad \mathsf{else} \ g(f(a \cdot c) \ \cup \ F(f,g)(a + a \cdot b,\, b,\, c)) \ \mathsf{fi} \ . &
\end{array}
$$

This terminates whenever the set of types in the underlying Kleene algebra is upward noetherian, i.e, has no infinite $\leq$-ascending chains. This is always the case in LAN,

whereas in REL and PAT it holds only if A is finite. A suitable termination measure is $a^\urcorner$, since

$$(a + a \cdot b)^\urcorner \;=\; a^\urcorner + (a \cdot b)^\urcorner \;\geq\; a^\urcorner$$

and

$$(a + a \cdot b)^\urcorner \;=\; a^\urcorner \;\Leftrightarrow\; (a \cdot b)^\urcorner \;\leq\; a^\urcorner \;.$$

Particular instances of this schematic algorithm compute are algorithms for reachability and shortest connecting paths, see [3].

In graph applications, the parameter a in the above algorithm carries all paths of the graph which have already been visited during the layered traversal from the starting set. The efficiency of the algorithm can be improved by introducing an additional parameter u that contains all already computed paths, while a carries only those paths whose last node has not been visited by any other path. This can again be done in the general framework of typed Kleene algebras (see [3] for details). The result is

$$\begin{array}{l}
F_{\mathit{eff}}(f,g)(a,b,c,u) = \\
\quad \mathsf{let}\ v \;=\; a + u \\
\qquad\quad b_1 \;=\; b \cdot v^\urcorner \\
\qquad\quad b_2 \;=\; b \backslash b_1 \\
\quad \mathsf{in}\ \ (a^\urcorner \sqcap u^\urcorner = 0) \cdot \\
\qquad\quad \mathsf{if}\ \ulcorner c \;\leq\; v^\urcorner \;\vee\; (v \cdot b)^\urcorner \;\leq\; v^\urcorner \\
\qquad\qquad \mathsf{then}\ g(f(v \cdot c)) \\
\qquad\qquad \mathsf{else}\ \ g(f(v \cdot c) + F_{\mathit{eff}}(f,g)(v \cdot b_2, b, c, v))\ \mathsf{fi}\ \ .
\end{array}$$

First we note that, for type x,

$$b \backslash (b \cdot x) \;=\; b \cdot \neg\, x \;, \tag{8}$$

where the negation $\neg\, x$ is the relative complement of x w.r.t. 1. Indeed,

$$\begin{array}{ll}
 & b \\
= & \quad \{\![\text{ neutrality }]\!\} \\
 & b \cdot 1 \\
= & \quad \{\![\text{ complement }]\!\} \\
 & b \cdot (x + \neg\, x) \\
= & \quad \{\![\text{ distributivity }]\!\} \\
 & b \cdot x + b \cdot \neg\, x
\end{array}$$

and

$$\begin{array}{ll}
 & b \cdot x \sqcap b \cdot \neg\, x \\
= & \quad \{\![\text{ typedness }]\!\} \\
 & b \cdot (x \sqcap \neg\, x) \\
= & \quad \{\![\text{ complement }]\!\} \\
 & b \cdot 0 \\
= & \quad \{\![\text{ strictness }]\!\} \\
 & 0 \;.
\end{array}$$

We now show that the recursive call preserves the invariant $a^{\urcorner} \sqcap u^{\urcorner} = 0$. To this end, set

$$\begin{aligned}
a' &\stackrel{\text{def}}{=} v \cdot b_2 \ , \\
b' &\stackrel{\text{def}}{=} b \ , \\
u' &\stackrel{\text{def}}{=} v \ , \\
v' &\stackrel{\text{def}}{=} a' + u' \ = \ v \cdot b_2 + v \ , \\
b'_1 &\stackrel{\text{def}}{=} b' \cdot v'^{\urcorner} \ , \\
b'_2 &\stackrel{\text{def}}{=} b' \backslash b'_1 \ = \ b \cdot \neg \ v'^{\urcorner} \ .
\end{aligned}$$

Then we have to show that $a'^{\urcorner} \sqcap u'^{\urcorner} = 0$ provided $a^{\urcorner} \sqcap u^{\urcorner} = 0 \wedge \neg \ (\ulcorner c \ \leq \ v^{\urcorner} \ \vee \ (v \cdot b)^{\urcorner} \ \leq \ v^{\urcorner})$.

We calculate

$$\begin{aligned}
& a'^{\urcorner} \sqcap u'^{\urcorner} \\
= \quad & \{\![\text{ definitions }]\!\} \\
& (v \cdot b_2)^{\urcorner} \sqcap v^{\urcorner} \\
= \quad & \{\![\text{ by (8) }]\!\} \\
& (v \cdot b \cdot \neg \ v^{\urcorner})^{\urcorner} \sqcap v^{\urcorner} \\
\leq \quad & \{\![\text{ by Lemma 33.2 }]\!\} \\
& (\neg \ v^{\urcorner})^{\urcorner} \sqcap v^{\urcorner} \\
= \quad & \{\![\text{ by Lemma 33.3 and } \neg \ v^{\urcorner} \text{ a type }]\!\} \\
& \neg \ v^{\urcorner} \sqcap v^{\urcorner} \\
= \quad & \{\![\text{ complement }]\!\} \\
& 0 \ .
\end{aligned}$$

Next we show that one can strengthen the invariant by the conjunct $u \cdot b_2 \leq v$. So we need to show $u' \cdot b'_2 \leq v'$ provided $u \cdot b_2 \leq v$. We calculate

$$\begin{aligned}
& u' \cdot b'_2 \\
= \quad & \{\![\text{ definitions }]\!\} \\
& v \cdot b \cdot \neg \ v'^{\urcorner} \\
= \quad & \{\![\text{ definitions }]\!\} \\
& v \cdot b \cdot \neg \ (v \cdot b_2 + v)^{\urcorner} \\
= \quad & \{\![\text{ distributivity }]\!\} \\
& v \cdot b \cdot \neg \ ((v \cdot b_2)^{\urcorner} + v^{\urcorner}) \\
= \quad & \{\![\text{ de Morgan }]\!\} \\
& v \cdot b \cdot (\neg \ (v \cdot b_2)^{\urcorner} \sqcap \neg \ v^{\urcorner}) \\
\leq \quad & \{\![\text{ infimum }]\!\} \\
& v \cdot b \cdot \neg \ v^{\urcorner} \\
= \quad & \{\![\text{ definitions }]\!\} \\
& v \cdot b_2 \\
= \quad & \{\![\text{ by idempotence of } +, \text{ since } v \ = \ a + u \ ]\!\} \\
& (v + u) \cdot b_2
\end{aligned}$$

$$
\begin{array}{ll}
= & \quad \{\![\text{ distributivity }]\!\} \\
& v \cdot b_2 + u \cdot b_2 \\
\leq & \quad \{\![\text{ assumption }]\!\} \\
& v \cdot b_2 + v \\
= & \quad \{\![\text{ definitions }]\!\} \\
& v' \ .
\end{array}
$$

With this stronger invariant the algorithm simplifies to

$$
\begin{array}{l}
F_{\mathit{eff2}}(f,g)(a,b,c,u) = \\
\quad \mathsf{let}\ v \ = \ a + u \\
\qquad b_1 \ = \ b \cdot v^{\urcorner} \\
\qquad b_2 \ = \ b \backslash b_1 \\
\quad \mathsf{in}\ \ (a^{\urcorner} \sqcap u^{\urcorner} = 0 \ \wedge \ u \cdot b_2 \leq v) \cdot \\
\qquad \mathsf{if}\ \lceil c \ \leq \ v^{\urcorner} \ \vee \ (a \cdot b_2)^{\urcorner} \ \leq \ v^{\urcorner} \\
\qquad\quad \mathsf{then}\, g(f(v \cdot c)) \\
\qquad\quad \mathsf{else}\ \ g(f(v \cdot c) + F_{\mathit{eff2}}(f,g)(a \cdot b_2, b, c, v))\ \mathsf{fi}\ \ .
\end{array}
$$

Here the expensive computation of $v \cdot b_2$ is reduced to that of $a \cdot b_2$.

Acknowledgements: I am grateful to T. Ehm for critical remarks on preliminary versions of this paper and to J. Desharnais for pointing out the counterexample to locality of composition.

References

1. A.V. Aho, J.E. Hopcroft, J.D. Ullman: The design and analysis of computer algorithms. Reading, Mass.: Addison Wesley 1974
2. D.N. Arden: Delayed logic and finite state machines. In: Theory of computing machine design. Univ. of Michigan Press, Ann Arbor 1960, 1–35
3. T. Brunn, B. Möller, M. Russling: Layered Graph Traversals and Hamiltonian Path Problems – An Algebraic Approach. In: J. Jeuring (ed.): Mathematics of Program construction. Lecture Notes in Computer Science **1422**. Berlin: Springer 1998, 96–121
4. J.H. Conway: Regular algebra and finite machines. London: Chapman and Hall 1971
5. R.M. Dijkstra: Computation calculus — bridging a formalization gap. In: J. Jeuring (ed.): Proc. MPC 1998. LNCS 1422, 151–174
6. N.D. Gautam: The validity of equations of complex algebras. Arch. Math. Logik Grundlag. **3**, 117–124 (1957)
7. S.C. Kleene: Introduction to metamathematics. New York: van Nostrand 1952
8. P. Lescanne: Modèles non déterministes de types abstraits. R.A.I.R.O. Informatique théorique **16**, 225–244 (1982)
9. B. Möller: Relations as a program development language. In: B. Möller (ed.): Constructing programs from specifications. Proc. IFIP TC2/WG 2.1 Working Conference on Constructing Programs from Specifications, Pacific Grove, CA, USA, 13–16 May 1991. Amsterdam: North-Holland 1991, 373–397
10. B. Möller: Derivation of graph and pointer algorithms. In: B. Möller, H.A. Partsch, S.A. Schuman (eds.): Formal program development. Lecture Notes in Computer Science **755**. Berlin: Springer 1993, 123–160
11. B. Möller: Assertions and recursions. In: G. Dowek, J. Heering, K. Meinke, B. Möller (eds.): Higher order algebra, logic and term rewriting. Second Inter-

national Workshop, Paderborn, Sept. 21-22, 1995. Lecture Notes in Computer Science **1074**. Berlin: Springer 1996, 163–184
12. B. Möller, M. Russling: Shorter paths to graph algorithms. In: R.S. Bird, C.C. Morgan, J.C.P. Woodcock (eds.): Mathematics of Program Construction. Lecture Notes in Computer Science **669**. Berlin: Springer 1993, 250–268. Extended version: Science of Computer Programming **22**, 157–180 (1994)
13. M. Russling: Deriving a class of layer-oriented graph algorithms. Science of Computer Programming **26**, 117-132 (1996).
14. M. Russling: Deriving general schemes for classes of graph algorithms. Augsburger Mathematisch-Naturwissenschaftliche Schriften, Band 13 (1996).
15. G. Schmidt, T. Ströhlein: Relations and graphs. Discrete Mathematics for Computer Scientists. EATCS Monographs on Theoretical Computer Science. Berlin: Springer 1993