MAX-PLANCK-INSTITUT FÜR INFORMATIK

Peirce Algebras

Chris Brink Katarina Britz
Renate A. Schmidt

MPI–I–92–229 July 1992

INFORMATIK

Im Stadtwald
W 6600 Saarbrücken
Germany

Authors' Addresses

Chris Brink
Department of Mathematics, University of Cape Town,
Rondebosch 7700, South Africa.
Email: cbrink@ucthpx.uct.ac.za.

Katarina Britz
Department of Mathematics, University of Cape Town,
Rondebosch 7700, South Africa.
Email: brtkat01@ucthpx.uct.ac.za.

Renate Schmidt
Max-Planck-Institut für Informatik, Im Stadtwald,
W-6600 Saarbrücken 11, Germany.
Email: Renate.Schmidt@mpi-sb.mpg.de.

Acknowledgements

We are indebted to Michael Böttner for his early contribution to this paper. For financial support we thank the South African Foundation for Research Development and the Deutsche Forschungsgemeinschaft, SFB 314 (TICS).

Abstract

We present a two-sorted algebra, called a *Peirce algebra*, of relations and sets interacting with each other. In a Peirce algebra, sets can combine with each other as in a Boolean algebra, relations can combine with each other as in a relation algebra, and in addition we have both a relation-forming operator on sets (the Peirce product of Boolean modules) and a set-forming operator on relations (a cylindrification operation). Two applications of Peirce algebras are given. The first points out that Peirce algebras provide a natural algebraic framework for modelling certain programming constructs. The second shows that the so-called *terminological logics* arising in knowledge representation have evolved a semantics best described as a calculus of relations interacting with sets.

Keywords

Boolean modules, relation algebras, terminological logics, weakest prespecifications.

1 Introduction

In its modern form the algebra of relations has been under investigation by mathematicians since Tarski's seminal (1941) paper. The main line of development has been the study of a class of algebras called *relation algebras* (see, e.g. Jónsson (1982)), in parallel with developments such as Boolean algebras with operators (Jónsson and Tarski (1951), (1952)) and cylindric algebras (Henkin, Monk and Tarski (1985)). Since the early seventies the algebra of relations has increasingly become of interest to computer scientists as well. Just as the notion of a partial function provides a natural model for deterministic programs, so the more general notion of a (binary) relation provides a natural model for nondeterministic programs. This idea has been exploited in Blikle (1987) and in Sanderson (1981); it is evident in Floyd-Hoare logic for program verification and has been extended to specification in Hoare and He Jifeng (1987); it has figured in logics of programs such as dynamic logic (Parikh (1981), Harel (1984)), and it was used in the early seventies to model recursive procedures (De Bakker and De Roever (1973), Hitchcock and Park (1972)). Recently the algebra of relations has been extensively used in a graph-theoretic approach to programs by Schmidt and Ströhlein (1991). The proof theory of relations is also of interest to computer scientists, and several natural deduction systems are available (Wadge (1975), Hennessey (1980), Maddux (1983)).

In many cases it has become clear that we need, not just an algebra of relations as distinct from an algebra of sets, but an algebra of relations *interacting* with sets. (For example, if we view a program as effecting a transition on a state space, we may wish to model this by a binary relation acting on a set of states.) Such an algebra was presented in Brink (1981) under the name of *Boolean modules*. Though independent of the computer science context, Boolean modules are very similar to *dynamic algebras*, introduced by Kozen (1980) as the algebraic version of dynamic logic. And both of these are quite similar to the *extended relation algebras* introduced by Suppes (1976) in a linguistic context. However, Boolean modules and dynamic algebras both have the drawback of not treating relations (programs) and sets equally: there is a set-forming operator on relations, but no relation-forming operator on sets. Extended relation algebras do not have this drawback, but they do have the drawback of being as yet unformalized as algebras.

We present here a two-sorted algebra, called a *Peirce algebra*, of relations and sets interacting with each other. Peirce algebras were first defined in Britz (1988). In a Peirce algebra, sets (or rather, the variables representing sets) can combine with each other as in a Boolean algebra, relations can combine with each other as in a relation algebra, and in addition we have both a set-forming operator on relations and a relation-forming operator on sets. The former is the *Peirce product* used in Boolean modules; the latter is the operation of *cylindrification*. Peirce algebras thus present a natural next step after Boolean algebras, relation algebras and Boolean modules. It is true that within a relation algebra one can mimic the behaviour of sets (by so-called *right ideal elements*), and that in a Boolean module one has some grasp on relations by the way in which they act on sets. But in a Peirce algebra one can actually manipulate both sets and relations simultaneously. From an applications-oriented point of view this is an advantage, and we present two (sets of) sample applications to substantiate this point. The first (in §4.1) shows how some programming constructs in the calculus of weakest prespecification of Hoare and He (1987) can be modelled naturally in Peirce algebras. The second (in §4.2) points out that the so-called *terminological logics* arising in knowledge representation have evolved a semantics best described as a calculus of relations interacting with sets.

2 Interactions Between Sets and Relations

Relation Algebras

Let U be some universal set, and R and S binary relations over U. Some familiar relation-forming operations on relations are:

(2.1) The Boolean operations, i.e. intersection $\cap$, union $\cup$ and complement $'$
(2.2) *Composition*: $R\,;S = \{(x,y)\,|\,(\exists z)[(x,z) \in R \;\&\; (z,y) \in S]\}$
(2.3) *Converse*: $R^{\smile} = \{(x,y)\,|\,(y,x) \in R\}$
(2.4) *Identity*: $Id = \{(x,x)\,|\,x \in U\}$.

Relation algebras attempt to capture in equational form the calculus of relations, with operations such as these. We adopt here the standard definition of relation algebras from Chin and Tarski (1951), as modified in Tarski (1955).

(2.5) **Definition** A *relation algebra* is an algebra $\mathcal{R} = (R, +, \cdot, ', 0, 1, ;, ^{\smile}, e)$ satisfying the following axioms for each $r, s, t \in R$:

R1 $(R, +, \cdot, ', 0, 1)$ is a Boolean algebra
R2 $r\,;(s\,;t) = (r\,;s)\,;t$
R3 $r\,;e = r = e\,;r$
R4 $r^{\smile\smile} = r$
R5 $(r+s)\,;t = r\,;t + s\,;t$
R6 $(r+s)^{\smile} = r^{\smile} + s^{\smile}$
R7 $(r\,;s)^{\smile} = s^{\smile}\,;r^{\smile}$
R8 $r^{\smile}\,;(r\,;s)' \leq s'$.

The *full* relation algebra $\mathcal{R}(U)$ over a set U is the power set algebra $\mathbf{2}^{U^2}$ over U endowed with the composition, converse and identity operations defined in (2.2) to (2.4) above. The operations of *left* and *right residuation* treated in Blyth and Janowitz (1972) can be defined equationally in a relation algebra. The left residual $r\backslash s$ and the right residual r/s of $r, s \in R$ are defined by

(2.6) $r\backslash s = (s'\,;r^{\smile})'$
(2.7) $r/s = (s^{\smile}\,;r')'$

respectively. Certain classes of elements in a relation algebra will be of special interest to us. Let r be an element in a relation algebra. Then we say that:

(2.8) r is a *equivalence* element iff $r = r^{\smile}$ and $r\,;r = r$

(2.9) r is a *right ideal* element iff $r\,;1 = r$
(2.10) r is a *left ideal* element iff $1\,;r = r$
(2.11) r is an *ideal* element iff $1\,;r\,;1 = r$.

(2.12) **Theorem** In any relation algebra $\mathcal{R}$ the following hold for any $r, s, t, u \in R$:

R9 $e^{\smile} = e, \quad 0^{\smile} = 0, \quad 1^{\smile} = 1$
R10 $r \leq s$ iff $r^{\smile} \leq s^{\smile}$
R11 $(r \cdot s)^{\smile} = r^{\smile} \cdot s^{\smile}, \quad r'^{\smile} = r^{\smile\prime}$
R12 $r\,;0 = 0 = 0\,;r, \quad 1\,;1 = 1$
R13 $r\,;(s+t) = r\,;s + r\,;t$
R14 if $r \leq s$ then $t\,;r \leq t\,;s$ and $r\,;t \leq s\,;t$
R15 $(r\,;s) \cdot t = 0$ iff $(r^{\smile}\,;t) \cdot s = 0$ iff $(t\,;s^{\smile}) \cdot r = 0$
R16 $(r\,;s) \cdot (t\,;u) \leq r\,;[(r^{\smile}\,;t) \cdot (s\,;u^{\smile})]\,;u$.
R17 If $r \leq e$ then r is an equivalence element.
R18 If s is a right ideal element then $r \cdot s = (s \cdot e)\,;r$.

Proof. R9–R17 are proved in Chin and Tarski (1951).
R18: Applying R14 to $s \cdot e \leq e$ and using R3 we obtain $(s \cdot e)\,;r \leq e\,;r = r$. Similarly $(s \cdot e)\,;r \leq s\,;r \leq s\,;1$. Since s is a right ideal element, $s\,;1 = s$ and hence $(s \cdot e)\,;r \leq s$. Therefore $(s \cdot e)\,;r \leq r \cdot s$. It remains to be shown that $r \cdot s \leq (s \cdot e)\,;r$. By R3, $r \cdot s = s \cdot r = (e\,;s) \cdot (e\,;r)$. Using R16 we get $r \cdot s \leq e\,;[(e^{\smile}\,;e) \cdot (s\,;r^{\smile})]\,;r$. Hence $r \cdot s \leq [e \cdot (s\,;r^{\smile})]\,;r$ by R3 and R9. Since $r^{\smile} \leq 1$ and using R14 we get $r \cdot s \leq [e \cdot (s\,;1)]\,;r$. Consequently $r \cdot s \leq (e \cdot s)\,;r = (s \cdot e)\,;r$ since $s\,;1 = s$. □

Since 0 and 1 are right ideal elements, and since it is known that the set of right ideal elements is closed under joins, meets and complements, it forms a Boolean algebra. Similarly, the sets of left ideal elements and ideal elements also form Boolean algebras. In general, a non-trivial algebra $\mathcal{A}$ is said to be *simple* iff the identity relation over A and the complete relation A^2 are the only two congruence relations over A. Jónsson and Tarski (1952, Theorem 4.10) show that for every non-trivial relation algebra $\mathcal{R}$ the following are equivalent:

(2.13) (i) $\mathcal{R}$ is simple.
(ii) $\mathcal{R}$ has exactly two distinct ideal elements, namely 0 and 1.
(iii) For every $r \in R$, $\quad r \neq 0 \quad \Rightarrow \quad 1\,;r\,;1 = 1$.

Boolean Modules

In many cases it has become clear that we need not just an algebra of relations distinct from an algebra of sets, but an algebra of relations interacting with sets. Some familiar set-forming operations on sets and relations are:

(2.14) *Domain*: $\mathrm{dom}(R) = \{x \mid (\exists y)[(x,y) \in R]\}$
(2.15) *Range*: $\mathrm{ran}(R) = \{y \mid (\exists x)[(x,y) \in R]\}$

(2.16) *Image*: $R\,``A = \{y \mid (\exists x)[(x, y) \in R \;\&\; x \in A]\}$
(2.17) *Peirce product*: $R\,{:}\,A = \{x \mid (\exists y)[(x, y) \in R \;\&\; y \in A]\}$.

Note that operations (2.14) - (2.16) are variants of the Peirce product, since:

(2.18) $\mathrm{dom}(R) = R\,{:}\,U$
(2.19) $\mathrm{ran}(R) = R^{\smile}\,{:}\,U$
(2.20) $R\,``A = R^{\smile}\,{:}\,A$.

An algebra of relations interacting with sets via Peirce product was introduced by Brink (1981).

(2.21) **Definition** (Brink (1988)) A *Boolean module* is a two-sorted algebra $\mathcal{M} = (\mathcal{B}, \mathcal{R}, :)$, where $\mathcal{B}$ is a Boolean algebra, $\mathcal{R}$ is a relation algebra and $:$ is a mapping $\mathcal{R} \times \mathcal{B} \longrightarrow \mathcal{B}$ written $r : a$ such that for any $r, s \in R$ and $a, b \in B$:

M1 $r : (a + b) = r : a + r : b$
M2 $(r + s) : a = r : a + s : a$
M3 $r : (s : a) = (r\,;s) : a$
M4 $e : a = a$
M5 $0 : a = 0$
M6 $r^{\smile} : (r : a)' \leq a'$.

The order of precedence among the operations is $'$ and $^{\smile}$, then $:$, $;$, $\cdot$ and $+$ in decreasing order. The *full* Boolean module $\mathcal{M}(U)$ over a set U is the Boolean module $(\mathcal{B}(U), \mathcal{R}(U), :)$, where $\mathcal{B}(U)$ is the power set algebra $\mathbf{2}^U$ over U, $\mathcal{R}(U)$ is the full relation algebra over U, and $:$ is the Peirce product defined in (2.17) above. An *ideal* element in a Boolean module is defined analogously to ideal elements in a relation algebra. It is any element a in the underlying Boolean algebra such that $1 : a = a$. The only simple Boolean algebra is $\{0, 1\}$. Brink (1981, Theorem 4.1) shows that for every non-trivial Boolean module $\mathcal{M}$ the following are equivalent:

(2.22) (i) $\mathcal{M}$ is simple.
(ii) $\mathcal{M}$ has exactly two distinct ideal elements, namely 0 and 1.
(iii) For every $a \in B$, $\quad a \neq 0 \;\Rightarrow\; 1 : a = 1$.

Dynamic Algebras

A dynamic algebra is a two-sorted algebra of programs and assertions, the algebra of programs being a *Kleene algebra* and the algebra of assertions being a Boolean algebra. Kleene algebras are closely related to relation algebras, while dynamic algebras are closely related to Boolean modules.

(2.23) **Definition** (Kozen (1981)) A *Kleene algebra* is an algebra $\mathcal{K} = (K, +, 0, ;, ^*, e)$ satisfying the following axioms for each $r, s, t \in K$:

K1 $(K, +, 0)$ is an upper semilattice
K2 $(K, ;, e)$ is a monoid
K3 $r\,; (s + t) = r\,; s + r\,; t$
K4 $(r + s)\,; t = r\,; t + s\,; t$

K5 $r\,;0 = 0\,;r = 0$
K6 $r\,;s^*\,;t = \sum_n (r\,;s^n\,;t)$

(2.24) **Definition** (Kozen (1981)) A *dynamic algebra* is a two-sorted algebra $\mathcal{D} = (\mathcal{B}, \mathcal{K}, f)$, where $\mathcal{B}$ is a Boolean algebra, $\mathcal{K}$ is a Kleene algebra and f is a mapping $\mathcal{K} \times \mathcal{B} \longrightarrow \mathcal{B}$ written $f(r, a) = f : a$ such that for any $r, s \in K$ and $a, b \in B$

D1 $r : (a + b) = r : a + r : b$
D2 $(r + s) : a = r : a + s : a$
D3 $r : (s : a) = (r\,; s) : a$
D4 $e : a = a$
D5 $0 : a = a : 0 = 0$
D6 $r^* : a = \sum_n (r^n : a)$.

This definition resembles that of Boolean modules, the difference being that Boolean modules are based on relation algebras (and hence have converse and complement operations but no Kleene closure), while dynamic algebras are based on Kleene algebras (and hence have a Kleene closure operation but no converses or complements). There is an extensive discussion of the relative merits of Boolean modules and dynamic algebras in Pratt (1990).

Both Boolean modules and dynamic algebras characterize *set-forming* operations algebraically, but do not address the problem of defining algebraically any *relation-forming* operation on relations and sets. This is the topic of the next section. Some familiar such relation-forming operations are:

(2.25) *Domain restriction*: $R \lceil A = \{(x, y) \mid (x, y) \in R \;\&\; x \in A\}$
(2.26) *Range restriction*: $R \rfloor A = \{(x, y) \mid (x, y) \in R \;\&\; y \in A\}$
(2.27) *Cartesian product*: $A \times B = \{(x, y) \mid x \in A \;\&\; y \in B\}$
(2.28) *Test operation*: $A? = \{(x, x) \mid x \in A\}$
(2.29) *Left cylindrification*: ${}^{c}A = \{(x, y) \mid y \in A\}$
(2.30) *Right cylindrification*: $A^c = \{(x, y) \mid x \in A\}$.

These operations are interdefinable. One can for example interdefine the operations given in (2.25) to (2.29) in terms of right cylindrification as follows:

(2.31) $R \lceil A = R \cap A^c, \qquad A^c = U^2 \lceil A$
(2.32) $R \rfloor A = R \cap A^{c\smile}, \qquad A^c = (U^2 \rfloor A)^{\smile}$
(2.33) $A \times B = A^c \cap B^{c\smile}, \qquad A^c = A \times U$
(2.34) $A? = A^c \cap Id, \qquad A^c = A?\,; U^2$
(2.35) ${}^{c}A = A^{c\smile}, \qquad A^c = ({}^{c}A)^{\smile}$.

3 Peirce Algebras

In order to characterize the relation-forming operations defined in (2.25) to (2.30), we extend the notion of a Boolean module to include a relation-forming operation on sets. Accordingly, we define a *Peirce algebra* to be a Boolean module $(\mathcal{B}, \mathcal{R}, :)$

enriched with an operation from the underlying Boolean algebra $\mathcal{B}$ to the underlying relation algebra $\mathcal{R}$. This operation is the algebraic counterpart to right cylindrification.

(3.1) **Definition** Let $\mathcal{B} = (B, +, \cdot, ', 0, 1)$ be a Boolean algebra and $\mathcal{R} = (R, +, \cdot, ', 0, 1, ;, \smile, e)$ be a relation algebra. A *Peirce algebra* is a two-sorted algebra $\mathcal{P} = (\mathcal{B}, \mathcal{R}, :, ^c)$, where $(\mathcal{B}, \mathcal{R}, :)$ is a Boolean module and $^c : B \longrightarrow R$ a mapping such that for every $a \in B$ and $r \in R$:

P1 $\quad a^c : 1 = a$

P2 $\quad (r : 1)^c = r ; 1$.

The assumed order of precedence is ^{c}, $'$ and $\smile$, then $:$, $;$, $\cdot$ and $+$ in descending order.

Examples of Peirce Algebras

(i) The full Boolean module $\mathcal{M}(U)$ over a non-empty set U endowed with right cylindrification is a Peirce algebra. For, let A be any subset of U and R any relation over U. Axiom P1 states that the domain of $A^c = A \times U$ is A, which is true. As for axiom P2, R composed with the universal relation U^2 is the set of (x, y) such that $x \in \mathrm{dom}(R)$ and $y \in U$, that is, $R ; U^2 = \mathrm{dom}(R) \times U$. Hence $R ; U^2 = (\mathrm{dom}(R))^c = (R : U)^c$ by (2.18) and (2.33). We call the Peirce algebra $(\mathcal{B}(U), \mathcal{R}(U), :, ^c)$ the *full Peirce algebra* over U and denote it by $\mathcal{P}(U)$.

(ii) Suppes (1976) introduced the notion of an *extended relation algebra*. In (1976, 1981) he defines an extended relation algebra over a non-empty set U as the family of sets $\mathbf{2}^U \cup \mathbf{2}^{U^2}$ closed under the set-theoretic operations union, intersection, complementation, relational composition, converse, image and domain restriction. Complementation on a set in $\mathbf{2}^U$ is taken with respect to U and complementation on a relation in $\mathbf{2}^{U^2}$ with respect to U^2. An extended relation algebra is denoted by $\mathcal{E}(U)$.

Extended relation algebras are used in the context of computational linguistics (Suppes (1976, 1979, 1981) and Böttner (1992a, 1992b)). They specify the model structure of certain fragments of natural language. For example, the sentence 'Some persons do not eat some foods' is interpreted by $P \cap (E^{\smile\prime}$ "$F) \neq 0$ where P, E and F denote the set of persons, the relation of eating and the set of foods, respectively (Suppes (1981, p. 406)).

Extended relation algebras are not algebras in the same sense as relation algebras, Boolean modules or Peirce algebras. They are more appropriately thought of as *calculi* of sets and relations interacting with each other. An extended relation algebra $\mathcal{E}(U)$ can be transformed into the full Peirce algebra $\mathcal{P}(U)$ by explicitly distinguishing between the operations on the sets in $\mathbf{2}^U$ and the operations on the relations in $\mathbf{2}^{U^2}$ and defining the image and domain restriction operations as in (2.20) and (2.31) in terms of Peirce product and right cylindrification, respectively.

(iii) Any relation algebra $\mathcal{R}$ can be regarded as a Peirce algebra $(\mathcal{B}, \mathcal{R}, ;, \iota)$, where $\mathcal{B}$ is the Boolean algebra of right ideal elements of $\mathcal{R}$, $;$ is the map $\mathcal{R} \times \mathcal{B} \longrightarrow \mathcal{B}$

satisfying M1 to M6 and ι the map $\mathcal{B} \longrightarrow \mathcal{R}$ defined by $\iota(r) = r$. The operation ; is the composition in $\mathcal{R}$ restricted to $\mathcal{B}$, and ι satisfies P1 and P2.

Arithmetic

The next theorem lists a number of arithmetical properties of Peirce algebras.

(3.2) **Theorem** In any Peirce algebra $(\mathcal{B}, \mathcal{R}, :, {}^c)$ the following hold for each $a, b \in B$ and $r, s \in R$.

P3	a^c is a right ideal element
P4	$0^c = 0, \qquad 1^c = 1$
P5	$(a+b)^c = a^c + b^c$
P6	$a'^c = a^{c\prime}$
P7	$(a \cdot b)^c = a^c \cdot b^c$
P8	$a = b$ iff $a^c = b^c$
P9	$a \leq b$ iff $a^c \leq b^c$
P10	$(r\,; a^c) : 1 = r : a$
P11	$(r : a)^c = r\,; a^c$
P12	$r : a = b$ iff $r\,; a^c = b^c$
P13	$r : 1 = 0$ iff $r = 0$
P14	$a^c \cdot e$ is an equivalence element
P15	$r \cdot a^c = (a^c \cdot e)\,; r, \qquad a^c = (a^c \cdot e)\,; 1$
P16	$r \cdot a^{c\smile} = r\,; (a^c \cdot e), \qquad a^{c\smile} = 1\,; (a^c \cdot e)$
P17	$(a^c \cdot e) : 1 = a$
P18	$(a^c \cdot e) : b = a \cdot b$
P19	$(a^c \cdot e) : a = a$
P20	$(a^c \cdot e) : a' = 0$
P21	$(r \cdot a^{c\smile}) : 1 = r : a$
P22	$(r \cdot a^{c\smile}) : b = r : (a \cdot b)$.

Proof. P3 follows from the axioms, since $a^c\,; 1 = (a^c : 1)^c = a^c$.
P4: $0 = 0 : 1$ by M5. Hence $0^c = (0 : 1)^c = 0\,; 1 = 0$ by M5, P2 and R12. Similar for $1^c = 1$.
P5: $(a+b)^c = ((a^c + b^c) : 1)^c$, since $a + b = a^c : 1 + b^c : 1 = (a^c + b^c) : 1$ by P1 and M2. Therefore $(a+b)^c = (a^c + b^c)\,; 1 = a^c\,; 1 + b^c\,; 1 = a^c + b^c$ by P2, R5 and P3.
P6: $a^c + a'^c = (a + a')^c = 0^c = 0$ by P5 and P4. Then a'^c must coincide with $a^{c\prime}$, the unique complement of a^c.
P7: We use P6 and P5 together with the De Morgan laws: $(a \cdot b)^c = (a' + b')'^c = (a' + b')^{c\prime} = (a'^c + b'^c)' = (a^{c\prime} + b^{c\prime})' = a^c \cdot b^c$.
P8: If $a = b$ then clearly $a^c = b^c$. Conversely, suppose $a^c = b^c$. Then $a^c : 1 = b^c : 1$, hence $a = b$ by P1.
P9: Using P8 and P5 we get $a + b = b$ iff $(a+b)^c = b^c$ iff $a^c + b^c = b^c$. Therefore $a \leq b$ iff $a^c \leq b^c$.
P10 follows by P1, since $(r\,; a^c) : 1 = r : (a^c : 1)$ by M3.

P11: Using P10, P2 and P3 we get $(r:a)^c = ((r;a^c):1)^c = r;a^c;1 = r;a^c$.
P12 follows by P8 and P11: $r:a = b$ iff $(r:a)^c = b^c$ iff $r;a^c = b^c$.
P13: $(r;1)\cdot 1 = 0$ iff $(1;1^{\smile})\cdot r = 0$ by R15. Then $r;1 = 0$ iff $r = 0$ since $1;1^{\smile} = 1$. Hence $(r:1)^c = 0^c$ iff $r = 0$ by P12 which implies that $r:1 = 0$ iff $r = 0$ by P8.
P14 is immediate by R17.
P15 follows immediately by R18 since a^c is a right ideal element by P3. To establish $a^c = (a^c \cdot e);1$ let $r = 1$.
P16: Using R4, R11, P15, R7 and R17 we get $r \cdot a^{c\smile} = (r \cdot a^{c\smile})^{\smile\smile} = (r^{\smile} \cdot a^{c\smile\smile})^{\smile} = (r^{\smile} \cdot a^c)^{\smile} = ((a^c \cdot e);r^{\smile})^{\smile} = r^{\smile\smile};(a^c \cdot e)^{\smile} = r;(a^c \cdot e)$. Let $r = 1$ then $a^{c\smile} = 1;(a^c \cdot e)$.
P17 follows by P8 since $((a^c \cdot e):1)^c = (a^c \cdot e);1 = a^c$ by P2 and P15.
P18: $((a^{c\smile} \cdot e):b)^c = (a^{c\smile} \cdot e);b^c = a^c \cdot b^c = (a \cdot b)^c$ by P11, P15 and P7. The result then follows by P8.
P19 and P20 are immediate by P18.
P21: $(r \cdot a^{c\smile}):1 = (r;(a^c \cdot e)):1$ by P16. Thus $(r \cdot a^{c\smile}):1 = r:((a^c \cdot e):1) = r:a$ by M3 and P17.
P22: Since $a^{c\smile} \cdot b^{c\smile} = (a \cdot b)^{c\smile}$ by R11 and P7, we get, using P21 repeatedly, $(r \cdot a^{c\smile}):b = (r \cdot a^{c\smile} \cdot b^{c\smile}):1 = (r \cdot (a \cdot b)^{c\smile}):1 = r:(a \cdot b)$. $\square$

P4–P7 imply that the set of right cylindrification elements forms a Boolean algebra. In fact, as we show in Theorem (3.6) below, this Boolean algebra coincides with the Boolean algebra of right ideal elements in the relation algebra $\mathcal{R}$.

P12 provides us with a translation between arithmetical properties in the relation algebra and properties in the Boolean module. In particular, it provides a translation between properties of composition and properties of Peirce product. It explains the apparent parallelism between the arithmetic of relation algebras and Boolean modules (observed by Brink (1981) and Pretorius (1990)).

P14–P20 characterize elements of the form $a^c \cdot e$, thus establishing properties of the test operation defined in (2.28).

P21 and P22 establish properties of Peirce product applied to range restriction.

A Peirce algebra is *simple* if its underlying Boolean module is simple. The next theorem states that this definition is equivalent to requiring that the underlying relation algebra is simple.

(3.3) **Theorem** Let $(\mathcal{B}, \mathcal{R}, :, ^c)$ be a Peirce algebra. Then $\mathcal{R}$ is a simple relation algebra iff $(\mathcal{B}, \mathcal{R}, :)$ is a simple Boolean module.

Proof. Assume $\mathcal{R}$ is simple and a is any non-zero element in B. Then $a^c \neq 0^c = 0$ by P8 and P4 which implies $1;a^c;1 = 1$ by (2.13). Applying P11 and P2 yields $1 = 1;a^c;1 = ((1:a)^c:1)^c$. Using P1 we get $1 = (1:a)^c$. Hence $1^c = (1:a)^c$ by P4 and $1:a = 1$ by P8 which implies $(\mathcal{B}, \mathcal{R}, :)$ is simple by (2.22).

Conversely, suppose $(\mathcal{B}, \mathcal{R}, :)$ is simple. Let r be any non-zero element in R. Then $r:1 \neq 0$ by P13 and $1:(r:1) = 1$ by (2.22). So $1^c = (1:(r:1))^c$ by P8 and $(1:(r:1))^c = 1;r;1$ by P11 and P2. Therefore $1^c = 1;r;1$, i.e. $1 = 1;r;1$ by P4. Consequently $\mathcal{R}$ is simple by (2.13). $\square$

(3.4) **Theorem** Let $\mathcal{P}$ be a Peirce algebra with an underlying simple Boolean module. Then for each $a, b \in B$:

P23 $\quad a \neq 0 \quad \Rightarrow \quad a^{c\smile} : 1 = 1$
P24 $\quad b \neq 0 \quad \Rightarrow \quad a^c ; b^c = a^c$
P25 $\quad b \neq 0 \quad \Rightarrow \quad a^c : b = a$
P26 $\quad b \neq 0 \quad \Rightarrow \quad (r ; a^c) : b = r : a.$

Proof. P23: Assume $a \neq 0$. By Theorem (2.22) $1 : a = 1$. Applying P12 and P4 yields $1 ; a^c = 1^c = 1$. Then $a^{c\smile} ; 1 = (1 ; a^c)^\smile = 1$ by R9 and R7 and $a^{c\smile} ; 1 = 1^c$ by P4. Hence $a^{c\smile} : 1 = 1$ by P12.
P24: Let $b \neq 0$. Then $b^c \neq 0^c = 0$. By Theorem (3.3) the relation algebra of $\mathcal{P}$ is simple. Hence $1 ; b^c ; 1 = 1$ by Theorem (2.13). Therefore $a^c ; 1 ; b^c ; 1 = a^c ; 1$. Using P3 we obtain the required property.
P25 follows directly from P24 by P12.
P26: $(r ; a^c) : b = r : (a^c : b) = r : a$ by M3 and P25 whenever $b \neq 0$. □

P23 and P25 respectively state that in a full Peirce algebra $\mathcal{P}(U)$ over a non-empty set U the range of A^c is U whenever $A \neq \emptyset$, and $A^c : B$ is A whenever $B \neq \emptyset$ (and as a special case that the domain of A^c is A).

Peirce Algebra and Relation Algebra

In example (iii) above we pointed out that any relation algebra can be regarded as a Peirce algebra. The relationship between the Boolean algebra $\mathcal{B}$ and the relation algebra $\mathcal{R}$ of a Peirce algebra $(\mathcal{B}, \mathcal{R}, :, {}^c)$ is further clarified in Theorems (3.6) and (3.8). We show that $\mathcal{B}$ can be embedded in $\mathcal{R}$ in two ways: as its Boolean algebra of right ideal elements, and as its Boolean algebra of subsets of the identity relation.

(3.5) **Lemma** Let $(\mathcal{B}, \mathcal{R}, :, {}^c)$ be a Peirce algebra. There is a bijection between B and the set $\{r \in R \,|\, r = r ; 1\}$ of right ideal elements of $\mathcal{R}$.

Proof. Define $g : B \longrightarrow \{r \in R \,|\, r = r ; 1\}$ by $g(a) = a^c$ and $h : \{r \in R \,|\, r = r ; 1\} \longrightarrow B$ by $h(r) = r : 1$. Both g and h are well-defined. We show that g is bijective by showing that h is the inverse of g. For any $a \in B$, $h(g(a)) = a^c : 1 = a$ by P1 and for any $r \in R$ such that $r = r ; 1$, $g(h(r)) = (r : 1)^c = r ; 1 = r$ by P2. Hence h is the inverse of g as required. □

(3.6) **Theorem** Let $(\mathcal{B}, \mathcal{R}, :, {}^c)$ be a Peirce algebra. $\mathcal{B}$ and $(\{r \in R \,|\, r = r ; 1\}, +, \cdot, ', 0, 1)$ are isomorphic.

Proof. Follows from the previous lemma and P4–P7. □

By mapping the Boolean elements to the converse of their right cylindrification, $\mathcal{B}$ can be embedded analogously in $\mathcal{R}$ as the Boolean algebra of left ideal elements. $\mathcal{B}$ can also be embedded in $\mathcal{R}$ as the Boolean algebra of elements $r \in R$ such that $r \leq e$.

(3.7) **Lemma** Let $(\mathcal{B}, \mathcal{R}, :, ^c)$ be a Peirce algebra. There is a bijection between B and the set $\{r \in R \,|\, r \leq e\}$.

Proof. We proceed similarly as for Lemma (3.5). Define $g : B \longrightarrow \{r \in R \,|\, r \leq e\}$ by $g(a) = a^c \cdot e$ and $h : \{r \in R \,|\, r \leq e\} \longrightarrow B$ by $h(r) = (r\,;1)\!:\!1$. Again g and h are well-defined. Using P15 and P1 we obtain $h(g(a)) \;=\; ((a^c \cdot e)\,;1)\!:\!1 \;=\; a^c\!:\!1$ $= a$ for each $a \in B$. Now consider $g(h(r)) \;=\; ((r\,;1)\!:\!1)^c \cdot e$ for any $r \in R$ such that $r \leq e$. By P2 and R12 $g(h(r)) \;=\; (r\,;1) \cdot e$. Letting $s = 1$, $t = e$ and $u = e$ in R16 we obtain $(r\,;1) \cdot e \;\leq\; r\,;r^{\smile}\,;e$. Since r is an equivalence element by R17, $(r\,;1) \cdot e \;\leq\; r\,;e$ by Definition (2.8). Thus $(r\,;1) \cdot e \;\leq r$ by R3. Also, $r = \; r\,;e$ $\leq\; r\,;1$ by R14 since $e \leq 1$. Then $r \cdot e \;\leq\; (r\,;1) \cdot e$ and $r \leq\; (r\,;1) \cdot e$ since $r \leq e$. Therefore $(r\,;1) \cdot e \;= r$, i.e., $g(h(r)) \;= r$. Consequently, h is the inverse of g, hence g is bijective. □

(3.8) **Theorem** Let $(\mathcal{B}, \mathcal{R}, :, ^c)$ be a Peirce algebra. $\mathcal{B}$ and $(\{r \in R \,|\, r \leq e\}, +, \cdot, ^-, 0, e)$ are isomorphic.

Proof. Follows from the previous lemma and P4– P7. □

These results reiterate the point made by Maddux (1990) that Peirce algebras are not a mathematical requisite for modelling the interactions between sets and relations, in the sense that these interactions can be modelled in relation algebras. For instance, as mentioned in Chin and Tarski (1951), although the domain and range of a relation cannot be expressed in a relation algebra (because they are sets), their properties can be expressed in a relation algebra in terms of right and left ideal elements. Consider for example the statement that relations R and S have the same domain. In a proper relation algebra over U this can be expressed by the equation $R\,;U^2 = S\,;U^2$. The property that $\mathrm{dom}(R) = \mathrm{dom}(S)$ iff $R\,;U^2 = S\,;U^2$ can be verified set-theoretically. However, and this is an important observation, it can also be derived equationally in the framework of Peirce algebras. For we have $r\!:\!1 \;=\; s\!:\!1$ iff $(r\!:\!1)^c \;=\; (s\!:\!1)^c$ by P8 which implies $r\!:\!1 \;=\; s\!:\!1$ iff $r\,;1 \;=\; s\,;1$ by P2. This illustrates our view that while relation algebras may suffice to model the interactions between sets and relations, Peirce algebras provide a more natural framework for doing so.

Peirce Algebra and Dynamic Algebra

Theorem (3.9) makes the relationship between Peirce algebras and dynamic algebras more precise:

(3.9) **Theorem** Every join-complete Peirce algebra is a dynamic algebra.

Proof. We first need to show that every join-complete relation algebra is a Kleene algebra. To see this, we define the Kleene closure operation for relation algebras by $r^* = \sum_n r^n$ (where $\sum$ indicates least upper bound). Axioms K1 to K6 of of Definition (2.23) are now all theorems in the arithmetic of relation algebras. To complete the proof, we need to show that axioms D1 to D6 hold in any Peirce algebra. D1 to D5 correspond to M1 to M5. To prove D6, let $(\mathcal{B}, \mathcal{R}, :, {}^c)$ be a Peirce algebra, and let $\mathcal{B}'$ be the Boolean algebra $\{r \in R \mid r = r\,;1\}$. It follows from Lemma (3.6) that $\mathcal{B}'$ and $\mathcal{B}$ are isomorphic under the map $a\,;1 \to a:1$. Hence $(\mathcal{B}, \mathcal{R}, :)$ is isomorphic to $(\mathcal{B}, \mathcal{R}, ;)$. D6 now becomes $r^*\,; a = \sum_n (r^n\,; a)$, which was proved by Chin and Tarski (1951). So if $\mathcal{R}$ is join-complete, then $(\mathcal{B}, \mathcal{R}, ;)$ is a dynamic algebra, and hence so is $(\mathcal{B}, \mathcal{R}, :)$. □

4 Applications

4.1 Weakest Prespecifications

We show that Peirce algebras provide a natural way of modelling three different concepts in logics of programs. This comes about through the two isomorphism theorems proved in Theorems (3.6) and (3.8). First, Hoare and He (1987) use right ideal elements to model conditional statements in logics representing programs as binary relations. Second, subsets of the identity relation are used to model a test operation (Parikh (1981)). Third, left ideal elements can be used to model the initialization of abstract data types as defined in Hoare, He and Sanders (1987).

In the calculus of weakest prespecifications of Hoare and He (1987), a condition is defined as a binary relation for which $B = B\,; U^2$. That is, conditions are right ideal elements in the calculus of relations. This definition is explained as follows: a condition in a programming language is written as a Boolean expression. This corresponds to a set of *states* in which the expression is true. Since there is a correspondence between the Boolean algebra of right ideal elements in the calculus of binary relations over U, and the Boolean algebra of subsets of U, a condition is modelled by some subset A of U transformed into a binary relation over U with domain A and codomain U.

The natural algebraic counterpart for the calculus of weakest prespecifications is therefore a Peirce algebra. Elements of the Boolean algebra correspond to conditions; they are transformed into relations by the operation ^{c}, which maps elements of the Boolean algebra to right ideal elements of the relation algebra. A guarded command in the calculus of weakest prespecifications is a relation of the form $B \cap P$, where B is a condition and P is the statement executed if the condition is satisfied. In a Peirce algebra, this is an element of the form $b^c \cdot p$, where b is an element of the Boolean algebra and p is an element of the relation algebra.

In the calculus of weakest prespecifications, specifications of programs are modelled by binary relations. The weakest prespecification of a program Q and a specification R, denoted by $Q \backslash R$, is a specification of a program P such that, when P is executed followed by Q, their sequential composition meets R. In the calculus of relations, it is defined equationally by $Q \backslash R = (R'\,; Q^{\smile})'$, coinciding with definition (2.6) of the left residual of r and q in a relation algebra. The algebraic properties of weakest prespecification mentioned in Hoare and He (1987) are thus properties of left residuation in relation algebras. The notion of a weakest postspecification coincides similarly with that of right residuation.

Other authors have concidered the test operation ?, defined as a relation-forming operation on sets (Parikh (1981)). Set-theoretically, $A? = A^c \cap Id$. In a Peirce algebra, the test operation is defined by $a? = a^c \cdot e$. Now $a?\,;p = a^c \cdot p$ by P15, so that the definition of the guarded commands of Hoare and He (1987) coincide with that of Parikh (1981).

A third relation-forming operation on sets, that of mapping sets to left ideal elements in a Peirce algebra, has an application in the initialization of abstract data types. In Hoare, He and Sanders (1987), weakest prespecifications are used in the refinement of abstract data types. An abstract data type is a triple $AT = (AI, A, AF)$, where AI is the initialization, AF the finalization, and A a set of operations over the data type. In general, the initialization of a data type is any binary relation over some universe U, with U^2, the universal relation, representing random initialization, Id, the identity relation, representing no initialization, left ideal elements representing non-deterministic initialization, and left ideal elements arising from singleton sets representing deterministic initialization. That is, non-deterministic and deterministic initialization that do not depend upon the initial state of the abstract data type before initialization, are modelled naturally by elements of the form $a^{c\smile}$ in a Peirce algebra, where a is an element of the Boolean algebra. In the case of deterministic initialization, a would be a singleton set of the Boolean algebra.

4.2 Knowledge Representation

Brink and Schmidt (1992) proposed an algebraic approach to *knowledge representation*, more specifically *terminological representation*. It was shown that the *terminological (representation) language* $\mathcal{ALC}$ of Schmidt-Schauß and Smolka (1991) can be captured in the context of Boolean modules. It was further shown that inference about information expressed in this language can be carried out equationally. In this section we use Peirce algebras to accomodate terminological languages even more expressive than $\mathcal{ALC}$.

Terminological representation languages are part of the knowledge representation system KL-ONE (Brachman and Schmolze (1985)), and its many successsors including NIKL, KRYPTON, BACK and others. For a survey of these, see, e.g., Woods and Schmolze (1992). Terminological languages are used to encode knowledge to be stored in the knowledge base of a KL-ONE based system (more specifically in the terminological component, or T-box, if the system is a hybrid representation system such as KRYPTON or BACK).

Terminological representation languages have two syntactic primitives, called *concepts* and *roles*. Concepts are usually interpreted as sets and roles as binary relations. Concepts are ordered in a taxonomy by the *subsumption* relation which is interpreted as the subset relation. This taxonomy is called the *concept taxonomy*. Similarly roles are ordered in a separate *role taxonomy*. Common concept-forming operators are and, or and not which are used respectively to represent the conjunction, disjunction and negation of concepts. For example, let Males and Heirs be concepts representing the set of males and the set of heirs to the throne, respectively. Then the concept (and Males Heirs) represents the set of male heirs to the throne, (or Males Heirs) represents the set of all males and all heirs to the throne and (not Males) represents the set of humans who are not male (provided we assume the universe of discourse is the set of humans). These operators can also be used to construct new roles. Other typical role-forming operators are inverse and compose. With these the relation 'is a child of', which coincides with the relation 'has as parent', can be represented as the role (inverse parent-of) where parent-of is a role denoting the relation 'is a parent of'. And the relation 'is a grandmother of' which coincides with the relation 'is a mother of a parent of' can be represented as (compose mother-of parent-of) where mother-of denotes the relation 'is a mother of'. Most terminological languages also have operators that take both concepts and roles as arguments. An example of such an operator is the concept-forming operator some. For example, let Princes represent the set of princes. Then (some mother-of Princes) is a concept description representing the set of mothers of (some) princes. An example of a role-forming operator applied to both concepts and roles is the restrict operator. This operator can be used, for example, to represent the relation 'has as son', that is, the relation 'is a parent of' with its range restricted to the set of males, as the role (restrict parent-of Males). Subsumption relationships between concepts and roles are expressed using the ' $\sqsubseteq$ ' symbol. Mutual subsumption relationships, called *equivalences*, are denoted with the ' $\doteq$ ' symbol. For example the statements:

(4.1) Charles is a prince.
(4.2) All princes are male heirs to the throne.
(4.3) Elizabeth is a mother of Charles.
(4.4) Everybody who has someone as a son is a parent of that person.

can be represented in terminological languages as the following subsumption relations:

(4.1)′ Charles $\sqsubseteq$ Princes
(4.2)′ Princes $\sqsubseteq$ (and Males Heirs)
(4.3)′ Elizabeth $\sqsubseteq$ (some mother-of Charles)
(4.4)′ has-son $\sqsubseteq$ parent-of, where has-son $\doteq$ (restrict parent-of Males).

The different terminological representation languages distinguish themselves by the operators they provide. The terminological language $\mathcal{ALC}$ of Schmidt-Schauß and Smolka (1991) is a language in the class of so-called *attributive (concept) description languages*. These languages have been extensively analysed and classified with respect to the complexity of computing subsumption relations in a series of recent papers (e.g. Schmidt-Schauß and Smolka (1991), Hollunder *et al* (1990) and Donini *et al* (1991)). As their name suggests attributive concept description languages have concept-forming operators but no role-forming operators. $\mathcal{ALC}$ also does not provide for the expression of role subsumption relationships. Here we consider the expressively more powerful terminological language $\mathcal{U}$ which Patel-Schneider introduced in (1987). $\mathcal{U}$ is one of the most powerful existing terminological languages having $\mathcal{ALC}$ as a sublanguage.

In this section we extend the idea of Brink and Schmidt (1992) and formalize the language $\mathcal{U}$ in the context of Peirce algebra.

To begin with we define the syntax of $\mathcal{U}$. Our presentation is in line with that of Schmidt-Schauß and Smolka (1991) and Brink and Schmidt (1992). The syntax we use for $\mathcal{U}$ is essentially that of Patel-Schneider (1987). The vocabulary of $\mathcal{U}$ consists of three disjoint sets of symbols: the set of primitive concepts, the set of primitive roles and the set of structural symbols. (Primitive concepts and roles are also referred to as being 'atomic' or 'generic'.) There are two designated concepts, the *top concept* $\top$ and the *bottom concept* $\bot$, and one designated role, the *identity role* self. The set of structural symbols contains the operators as well as the symbols '$\sqsubseteq$' and '$\doteq$', which we define below. Let 'A' denote any primitive concept and 'C', 'D', ... any concept description constructed from other concepts and roles according to the following syntax rule (in Backus Naur Form):

$$(4.5)\qquad \mathsf{C},\mathsf{D} \longrightarrow \begin{array}{l} \mathsf{A} \mid (\mathsf{and\ C\ D}) \mid (\mathsf{or\ C\ D}) \mid (\mathsf{not\ C}) \mid \\ (\mathsf{some\ R\ C}) \mid (\mathsf{all\ R\ C}) \mid \\ (\mathsf{atleast}\ n\ \mathsf{R}) \mid (\mathsf{atmost}\ n\ \mathsf{R}) \mid \\ (\mathsf{rvm\ R\ S}) \mid (\mathsf{sd\ C\ Rb}_1 \ldots \mathsf{Rb}_k), \end{array}$$

where k is a positive integer, n is a non-negative integer and 'R' and 'S' denote role descriptions (and are defined in (4.7) below). The 'Rb_i' in the sd-construct denote so-called *role bindings* and have one of two forms. Namely:

$$(4.6)\qquad \mathsf{Rb}_i \longrightarrow (\subseteq \mathsf{R\ S}) \mid (\supseteq \mathsf{R\ S}).$$

Let 'Q' denote any primitive role. The role descriptions 'R', 'S', ... are defined according to the rule:

$$(4.7)\qquad \mathsf{R},\mathsf{S} \longrightarrow \begin{array}{l} \mathsf{Q} \mid (\mathsf{and\ R\ S}) \mid (\mathsf{or\ R\ S}) \mid (\mathsf{not\ R}) \mid \\ (\mathsf{inverse\ R}) \mid (\mathsf{compose\ R\ S}) \mid (\mathsf{trans\ R}) \mid \\ (\mathsf{restrict\ R\ C}). \end{array}$$

The model-theoretic semantics of $\mathcal{U}$ can be given similar to that of $\mathcal{ALC}$ as an *interpretation* $\mathcal{I}$ which is defined as the pair $(\mathcal{D}^{\mathcal{I}}, \cdot^{\mathcal{I}})$. $\mathcal{D}^{\mathcal{I}}$ denotes a set called the *domain* (or *universe*) of interpretation and $\cdot^{\mathcal{I}}$ is a map, called the *interpretation function*, which assigns to every concept description C a subset $\mathsf{C}^{\mathcal{I}}$ of $\mathcal{D}^{\mathcal{I}}$ and to every role R a binary relation $\mathsf{R}^{\mathcal{I}}$ over $\mathcal{D}^{\mathcal{I}}$. Applied to concept descriptions this assignment is constrained by the following conditions:

$$(4.8)\qquad \begin{array}{rcl} \top^{\mathcal{I}} & = & \mathcal{D}^{\mathcal{I}} \\ \bot^{\mathcal{I}} & = & \emptyset \\ (\mathsf{and\ C\ D})^{\mathcal{I}} & = & \mathsf{C}^{\mathcal{I}} \cap \mathsf{D}^{\mathcal{I}} \\ (\mathsf{or\ C\ D})^{\mathcal{I}} & = & \mathsf{C}^{\mathcal{I}} \cup \mathsf{D}^{\mathcal{I}} \\ (\mathsf{not\ C})^{\mathcal{I}} & = & (\mathsf{C}^{\mathcal{I}})' \qquad (= \mathcal{D}^{\mathcal{I}} - \mathsf{C}^{\mathcal{I}}) \\ (\mathsf{some\ R\ C})^{\mathcal{I}} & = & \left\{x \mid (\exists y)[(x,y) \in \mathsf{R}^{\mathcal{I}} \ \&\ y \in \mathsf{C}^{\mathcal{I}}]\right\} \\ (\mathsf{all\ R\ C})^{\mathcal{I}} & = & \left\{x \mid (\forall y)[(x,y) \in \mathsf{R}^{\mathcal{I}} \Rightarrow y \in \mathsf{C}^{\mathcal{I}}]\right\} \\ (\mathsf{atleast}\ n\ \mathsf{R})^{\mathcal{I}} & = & \left\{x \mid \mathrm{card}(\{y \mid (x,y) \in \mathsf{R}^{\mathcal{I}}\}) \geq n\right\} \\ (\mathsf{atmost}\ n\ \mathsf{R})^{\mathcal{I}} & = & \left\{x \mid \mathrm{card}(\{y \mid (x,y) \in \mathsf{R}^{\mathcal{I}}\}) \leq n\right\} \\ (\mathsf{rvm\ R\ S})^{\mathcal{I}} & = & \left\{x \mid (\forall y)[(x,y) \in \mathsf{R}^{\mathcal{I}} \Rightarrow (x,y) \in \mathsf{S}^{\mathcal{I}}]\right\} \\ (\mathsf{sd\ C\ Rb}_1 \ldots \mathsf{Rb}_k)^{\mathcal{I}} & = & \left\{x \mid (\exists y)[(x,y) \in \bigcap_{i=1}^{k} \mathsf{Rb}_i^{\mathcal{I}} \ \&\ y \in \mathsf{C}^{\mathcal{I}}]\right\}. \end{array}$$

(For any set A, $\mathrm{card}(A)$ denotes the cardinality of A.) The semantics of the role binding constructs Rb_i is given by:

$$
\begin{aligned}
(4.9)\qquad (\subseteq \mathsf{R}\ \mathsf{S})^{\mathcal{I}} &= \big\{(x,y) \mid (\forall z)[(x,z) \in \mathsf{R}^{\mathcal{I}} \Rightarrow (y,z) \in \mathsf{S}^{\mathcal{I}}]\big\} \\
(\supseteq \mathsf{R}\ \mathsf{S})^{\mathcal{I}} &= \big\{(x,y) \mid (\forall z)[(y,z) \in \mathsf{S}^{\mathcal{I}} \Rightarrow (x,z) \in \mathsf{R}^{\mathcal{I}}]\big\}.
\end{aligned}
$$

The role-forming constructs are interpreted according to the following conditions:

$$
\begin{aligned}
(4.10)\qquad \mathsf{self}^{\mathcal{I}} &= \big\{(x,x) \mid x \in \mathcal{D}^{\mathcal{I}}\big\} \\
(\mathsf{and}\ \mathsf{R}\ \mathsf{S})^{\mathcal{I}} &= \mathsf{R}^{\mathcal{I}} \cap \mathsf{S}^{\mathcal{I}} \\
(\mathsf{or}\ \mathsf{R}\ \mathsf{S})^{\mathcal{I}} &= \mathsf{R}^{\mathcal{I}} \cup \mathsf{S}^{\mathcal{I}} \\
(\mathsf{not}\ \mathsf{R})^{\mathcal{I}} &= (\mathsf{R}^{\mathcal{I}})' \qquad (= (\mathcal{D}^{\mathcal{I}} \times \mathcal{D}^{\mathcal{I}}) - \mathsf{R}^{\mathcal{I}}) \\
(\mathsf{inverse}\ \mathsf{R})^{\mathcal{I}} &= \big\{(x,y) \mid (y,x) \in \mathsf{R}^{\mathcal{I}}\big\} \\
(\mathsf{compose}\ \mathsf{R}\ \mathsf{S})^{\mathcal{I}} &= \big\{(x,y) \mid (\exists z)[(x,z) \in \mathsf{R}^{\mathcal{I}}\ \&\ (z,y) \in \mathsf{S}^{\mathcal{I}}]\big\} \\
(\mathsf{trans}\ \mathsf{R})^{\mathcal{I}} &= \mathsf{R}^{\mathcal{I}} \cup \bigcup_{k \geq 1} \big\{(x,y) \mid (\exists z_1) \ldots (\exists z_k)[(x,z_1) \in \mathsf{R}^{\mathcal{I}} \\
&\qquad \&\ \forall (1 \leq i < k)[(z_i, z_{i+1}) \in \mathsf{R}^{\mathcal{I}}]\ \&\ (z_k, y) \in \mathsf{R}^{\mathcal{I}}]\big\} \\
(\mathsf{restrict}\ \mathsf{R}\ \mathsf{C})^{\mathcal{I}} &= \big\{(x,y) \mid [(x,y) \in \mathsf{R}^{\mathcal{I}}\ \&\ y \in \mathsf{C}^{\mathcal{I}}]\big\}.
\end{aligned}
$$

Therefore, concepts are interpreted as sets and roles together with the role binding constructs are interpreted as binary relations. In particular, concepts arising from other concepts do so by the usual Boolean operations of intersection, union and complement. Concepts arising from other concepts and roles do so by set-forming operations on sets and relations. For example, the expression (some R C) is interpreted as the Peirce product $\mathsf{R}^{\mathcal{I}} : \mathsf{C}^{\mathcal{I}}$. With the exception of the number restriction operators atleast and atmost, the remaining concept-forming operators all, rvm and sd can be shown to be interpreted as variants of Peirce product. The designated role self is interpreted as the identity relation. Roles arising from other roles with the operators and, or, not, inverse, compose, trans are respectively interpreted by the operations intersection, union, complement, converse, composition and transitive closure. (Note that the transitive closure of a relation R is also given by $\bigcup_{n=1}^{\infty} R^n$ where $R^1 = R$ and $R^{n+1} = R\,;R^n$ for $n \geq 1$.) The role binding constructs are special roles and their semantics can be expressed in terms of residuation or equivalently as variants of composition. The restrict operator, an operator on concepts and roles yielding roles, is interpreted as range restriction which is definable in terms of left cylindrification. We refer to the language $\mathcal{U}$ without the operators atleast and atmost as $\mathcal{U}^-$. In column two of Table 1 we summarize the semantics of $\mathcal{U}^-$ as reformulated in terms of operations in the calculus sets and relations interacting with each other. To emphasize this context, we use the notation of Section 2 and abbreviate $\mathsf{C}^{\mathcal{I}}$, $\mathsf{D}^{\mathcal{I}}$, ... and $\mathsf{R}^{\mathcal{I}}$, $\mathsf{S}^{\mathcal{I}}$, ... by C, D, ... and R, S, ..., respectively. U abbreviates $\mathcal{D}^{\mathcal{I}}$ and Rb_i abbreviates $\mathsf{Rb}_i{}^{\mathcal{I}}$.

For expository purposes and without changing the expressive and deductive capabilities of $\mathcal{U}$ we have slightly adapted the original vocabulary in Patel-Schneider (1987). The names of some of the operators have been changed; and, or and compose are here defined as binary operators; for some we use the more general definition given by Schmidt-Schauß and Smolka (1991) (there denoted by $\exists R : C$), and we

Table 1: Algebraic Semantics of $\mathcal{U}^-$

	Terminological expression	Interpretation	Algebraic term
(i)	$\top$	U	1
	$\bot$	$\emptyset$	0
	(and C D)	$C \cap D$	$a \cdot b$
	(or C D)	$C \cup D$	$a + b$
	(not C)	C'	a'
(ii)	self	Id	e
	(and R S)	$R \cap S$	$r \cdot s$
	(or R S)	$R \cup S$	$r + s$
	(not R)	R'	r'
	(inverse R)	$R^{\smile}$	$r^{\smile}$
	(compose R S)	$R\,;S$	$r\,;s$
	(trans R)	$\bigcup_{n=1}^{\infty} R^n$	$\sum_{n=1}^{\infty} r^n$
	($\subseteq$ R S)	$(R\,;S^{\smile\prime})' = R^{\smile}\backslash S^{\smile}$	$(r\,;s^{\smile\prime})' = r^{\smile}\backslash s^{\smile}$
	($\supseteq$ R S)	$(R'\,;S^{\smile})' = R/S$	$(r'\,;s^{\smile})' = r/s$
(iii)	(some R C)	$R : C$	$r : a$
	(all R C)	$(R : C')'$	$(r : a')'$
	(rvm R S)	$((R \cap S') : U)'$	$((r \cdot s') : 1)'$
	(sd C $\mathsf{Rb}_1 \ldots \mathsf{Rb}_k$)	$(\bigcap_{i=1}^{k} Rb_i) : C$	$(\prod_{i=1}^{k} r_i) : a$
(iv)	(restrict R C)	$R \cap C^{c\smile} = R \rfloor C$	$r \cdot a^{c\smile} = r \rfloor a$

include the designated concepts $\top$ and $\bot$ in the vocabulary of $\mathcal{U}$. The some operator as defined by Patel-Schneider (1987) is applied only to a role R. Its interpretation is the following:

$$(4.11)\quad (\mathsf{some\ R})^{\mathcal{I}} \;=\; \left\{x \mid (\exists y)[(x,y) \in \mathsf{R}^{\mathcal{I}}]\right\}$$

which coincides with $\mathsf{R}^{\mathcal{I}} : \mathcal{D}^{\mathcal{I}} = \mathrm{dom}(\mathsf{R}^{\mathcal{I}})$. It is not difficult to prove that our version of $\mathcal{U}$ is equivalent to that of Patel-Schneider.

To complete the definition of $\mathcal{U}$ we define *subsumption* and *equivalence* by which concepts and roles are related. Syntactically, these relationships are expressed as *terminological axioms*. Terminological axioms are used when computing information implicit in the knowledge base. They are denoted by 'σ', 'τ', ... and take the form of *specializations* $\sqsubseteq$ (often also called subsumptions) or *equivalences* $\doteq$ and are defined by:

$$(4.12)\quad \sigma, \tau \quad \longrightarrow \quad \mathsf{C} \sqsubseteq \mathsf{D} \mid \mathsf{C} \doteq \mathsf{D} \mid \mathsf{R} \sqsubseteq \mathsf{S} \mid \mathsf{R} \doteq \mathsf{S}.$$

A set of terminological axioms is referred to as a *terminology* T. A terminology can be viewed as a presentation of a knowledge base.

An interpretation $\mathcal{I}$ *satisfies* (or *models*) a terminological axiom σ, written $\models_{\mathcal{I}} \sigma$, if and only if depending on the form of σ the following holds:

$$(4.13)\quad \begin{array}{lll} \models_{\mathcal{I}} \mathsf{A} \sqsubseteq \mathsf{B} & \text{iff} & \mathsf{A}^{\mathcal{I}} \subseteq \mathsf{B}^{\mathcal{I}} \\ \models_{\mathcal{I}} \mathsf{A} \doteq \mathsf{B} & \text{iff} & \mathsf{A}^{\mathcal{I}} = \mathsf{B}^{\mathcal{I}}, \end{array}$$

for A and B either both concepts or both roles. More generally, an interpretation $\mathcal{I}$ is a model for a terminology T, written $\models_{\mathcal{I}} T$, if and only if *every* terminological axiom in T is satisfied by $\mathcal{I}$. A terminological axiom σ is *entailed by* a terminology T, written $T \models \sigma$, if and only if σ is satisfied by *every* model of T. *Subsumption* and *equivalence* with respect to a terminology T, respectively denoted by '$\preceq_T$' and '$\approx_T$', are defined by:

$$(4.14)\quad \begin{array}{lll} \mathsf{A} \preceq_T \mathsf{B} & \text{iff} & T \models \mathsf{A} \sqsubseteq \mathsf{B} \\ \mathsf{A} \approx_T \mathsf{B} & \text{iff} & T \models \mathsf{A} \doteq \mathsf{B} \end{array}$$

where A and B are either both concepts or both roles. Subsumption and equivalence with respect to the empty terminology are denoted by $\preceq$ and $\approx$, respectively. Quotienting the set of concept descriptions with respect to semantic equivalence $\approx_T$ yields a poset ordered with respect to $\preceq_T / \approx_T$. This poset is called the *concept taxonomy*. The *role taxonomy* is defined similarly.

With specialization and equivalence relationships respectively interpreted as a subset relation and an equality, we have shown that the semantics of $\mathcal{U}^-$ can be accomodated in the calculus of sets and relations. Therefore, since this calculus is formalized in Peirce algebra, so is the semantics of $\mathcal{U}^-$. In particular, each concept (interpreted as a set) can be interpreted as an element in the underlying Boolean algebra and each role (interpreted as a binary relation) as an element in the underlying relation algebra. The algebraic interpretation of $\mathcal{U}^-$ is summarized in column three of Table 1 which lists the algebraic terms with which the different kinds of terminological expressions are associated. Observe that to accomodate the trans operator the underlying relation algebra needs to be join-complete (i.e. closed with respect to arbitrary joins). Table 1 is subdivided as follows:

(i) lists the designated concepts and the concept-forming operators, their interpretation in the calculus of sets and the corresponding interpretation in Boolean algebra.

(ii) lists the designated role and roles arising from other roles, their interpretation in the calculus of relations and the formalization in relation algebra.

(iii) lists concepts arising from concepts as well as roles. These constructs are interpreted as interactions between sets and relations through Peirce product and are catered for in Boolean modules.

(iv) lists the role-forming operator restrict on concepts which is interpreted with left cylindrification and is formalized in Peirce algebra.

It is not difficult to show that each universal identity in a Peirce algebra determines a semantic equivalence ($\approx$) true in any terminology. (By a universal identity we mean an equational axiom or property of any Peirce algebra.)

An alternative approach to computing subsumption and equivalence relations is then the algebraic approach. This approach uses equational reasoning. New subsumption and equivalence relationships are deduced equationally from the terminological axioms and the axioms of Peirce algebra. We do not elaborate on the equational approach as detailed presentations with worked out examples can be found in Brink and Schmidt (1992) as well as Schmidt (1991).

In conclusion we claim for the application of Peirce algebras to terminological representation and reasoning the following advantages:

(i) Peirce algebras provide a formal mathematical framework. By and large, the development of KL-ONE type knowledge representation and terminological representation has been implementation-driven and rather *ad hoc*, see Woods and Schmolze (1992). (Only recently research has started to focus on formal aspects such as semantics and tractability.) It seems a common belief that the structural description operator sd cannot be discarded because it cannot be defined using other terminological constructs, see Woods and Schmolze (1992, §5.2.1). However analysis in the context of Peirce algebra reveals that this is not true. From the algebraic translations of sd expressions (see Table 1) it is apparent that sd can be defined in terms of the some operator and other operators. In fact, Schmidt (1991) showed that sd and some are interdefinable. Thus, the operator sd is redundant in $\mathcal{U}$.

(ii) Peirce algebras are quite powerful. They are sufficiently expressive to cater for $\mathcal{U}^-$, a very expressive sublanguage of $\mathcal{U}$. With the exception of the atleast and atmost constructs most existing terminological constructs can be expressed in $\mathcal{U}^-$ and are thus also catered for in Peirce algebras. Examples of constructs not explicitly included in the language $\mathcal{U}^-$ (or the language $\mathcal{U}$) but expressible in $\mathcal{U}^-$ (or $\mathcal{U}$) are the top role, the bottom role and the roles (domain C) and (range C). The top and bottom roles are respectively interpreted as the universal relation $(\mathcal{D}^\mathcal{I})^2$ and the empty relation and are therefore respectively associated with the unit and the zero in the relation algebra. The domain and range operators are used in NIKL (see Schmolze (1989)) and the language $\mathcal{KL}$ defined in Woods and Schmolze

(1992). In the calculus, and hence also algebraically, ($\mathsf{domain}\ \mathsf{C}$) is interpreted as the left cylindrification of the set $\mathsf{C}^{\mathcal{I}}$, while ($\mathsf{range}\ \mathsf{C}$) is interpreted as the right cylindrification of $\mathsf{C}^{\mathcal{I}}$. An example of an operator not captured in Peirce algebra is a generalized version of the atleast operator, called the fillers operator, defined in Patel-Schneider (1989).

(iii) Peirce algebras are easy to use. Even the most complicated terminological constructs in $\mathcal{U}^-$ have straightforward algebraic translations (as summarized in Table 1). Unlike the model-theoretic interpretation the algebraic interpretation of terminological expressions is free of individual variables and quantifiers. The algebraic language thus provides a compact and elegant formalization for first-order statements and terminological expressions. Peirce algebras provide a natural axiomatization for reasoning with concepts and roles. The axiomatization is equational. As a consequence, terminological inferences are straightforward to compute from the terminological axioms and the algebraic ones. For examples and case studies refer to Brink and Schmidt (1992) and Schmidt (1991).

(iv) Terminological representation can be linked to other areas of application of Peirce algebra. In particular, we believe terminological representation can benefit from Suppes' (1976, 1979, 1981) and Böttner's (1992a, 1992b) linguistic analysis of English language sentences, referred to earlier. This work can be utilized for translating information expressed in ordinary English into terminological expressions, and *vice versa*.

References

Blikle, A. (1987), *MetaSoft Primer*, Vol. 288 of *Lecture Notes in Computer Science*, Springer-Verlag.

Blyth, T. S. and Janowitz, M. F. (1972), *Residuation Theory*, Pergamon Press, Oxford, England.

Böttner, M. (1992a), State transition semantics, *Theoretical Linguistics*. To appear.

Böttner, M. (1992b), Variable-free semantics for anaphora, *Journal of Philosophical Logic*. To appear.

Brachman, R. J. and Schmolze, J. G. (1985), An overview of the KL-ONE knowledge representation system, *Cognitive Science* **9**(2), 171–216.

Brink, C. (1981), Boolean modules, *Journal of Algebra* **71**(2), 291–313.

Brink, C. (1988), On the application of relations, *South African Journal of Philosophy* **7**(2), 105–112.

Brink, C. and Schmidt, R. A. (1992), Subsumption computed algebraically, *Computers and Mathematics with Applications* **23**, 329–342.

Britz, K. (1988), *Relations and programs*, Master's thesis, Department of Computer Science, University of Stellenbosch, Stellenbosch, South Africa.

Chin, L. H. and Tarski, A. (1951), Distributive and modular laws in the arithmetic of relation algebras, *University of California Publications in Mathematics* **1**(9), 341–384.

De Bakker, J. W. and De Roever, W. P. (1973), A calculus for recursive program schemes, *in* M. Nivat (ed.), *Symposium on Automata, Formal Languages and Programming*, North Holland, Amsterdam.

Donini, F. M., Lenzerini, M., Nardi, D. and Nutt, W. (1991), The complexity of concept languages, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, Morgan Kaufmann, San Mateo, California, pp. 151–162.

Harel, D. (1984), Dynamic logic, *in* D. Gabbay and F. Guenthner (eds), *Handbook of Philosophical Logic*, Vol. II, Reidel, Dordrecht, Holland, pp. 497–604.

Henkin, L., Monk, J. D. and Tarski, A. (1985), *Cylindric Algebras I*, Vol. 64 of *Studies in Logic and the Foundations of Mathematics*, North-Holland, Amsterdam.

Hennessey, M. C. B. (1980), A proof system for the first-order relational calculus, *Journal of Computer and System Sciences* **20**, 96–110.

Hitchcock, P. and Park, D. (1972), Induction rules and termination proofs, *in* M. Nivat (ed.), *Automata, Languages and Programming*, North-Holland, Amsterdam.

Hoare, C. A. R. and Jifeng, H. (1987), The weakest prespecification, *Information Processing Letters* **24**, 127–132.

Hoare, C. A. R., Jifeng, H. and Sanders, J. W. (1987), Prespecification in data refinement, *Information Processing Letters* **25**, 71–76.

Hollunder, B., Nutt, W. and Schmidt-Schauß, M. (1990), Subsumption algorithms for concept description languages, *Proceedings of the 9th European Conference on Artificial Intelligence*, pp. 348–353.

Jónsson, B. (1982), Varieties of relation algebras, *Algebra Universalis* **15**, 273–298.

Jónsson, B. and Tarski, A. (1951), Boolean algebras with operators, Part I, *American Journal of Mathematics* **73**, 891–939.

Jónsson, B. and Tarski, A. (1952), Boolean algebras with operators, Part II, *American Journal of Mathematics* **74**, 127–162.

Kozen, D. (1980), A representation theorem for models of $*$-free PDL, *in* J. de Bakker and J. van Leeuwen (eds), *Automata, Languages and Programming*, Vol. 85 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 351–36.

Kozen, D. (1981), On the duality of dynamic algebras and Kripke models, *in* E. Engeler (ed.), *Logic of Programs*, Vol. 125 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 1–11.

Maddux, R. D. (1983), A sequent calculus for relation algebras, *Annals of Pure and Applied Logic* **25**, 73–101.

Maddux, R. D. (1990). Personal communication with C. Brink.

Parikh, D. (1981), Propositional dynamic logic of programs: A survey, *in* E. Engeler (ed.), *Logic of Programs*, Vol. 125 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 102–144.

Patel-Schneider, P. F. (1987), *Decidable, Logic-Based Knowledge Representation*, PhD thesis, University of Toronto.

Patel-Schneider, P. F. (1989), A four-valued semantics for terminological logics, *Artificial Intelligence* **38**, 319–351.

Pratt, V. R. (1990), Dynamic algebras as a well-behaved fragment of relation algebras, *in* C. H. Bergman, R. D. Maddux and D. L. Pigozzi (eds), *Algebraic Logic and Universal Algebra in Computer Science*, Vol. 425 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 77–110.

Pretorius, J. P. G. (1990), *The algebra and topology of Boolean modules*, Master's thesis, Department of Mathematics, University of Cape Town, Cape Town, South Africa.

Sanderson, J. G. (1981), *A Relational Theory of Computing*, Vol. 82 of *Lecture Notes in Computer Science*, Springer-Verlag.

Schmidt, G. and Ströhlein, T. (1991), *Relations and Graphs*, Springer-Verlag, Berlin-Heidelberg.

Schmidt, R. A. (1991), *Algebraic terminological representation*, Master's thesis, Department of Mathematics, University of Cape Town, Cape Town, South Africa. Available as Thesis-Reprint TR 011. Also as Technical Report MPI-I-91-216, Max-Planck-Institut für Informatik, Saarbrücken, Germany.

Schmidt-Schauß, M. and Smolka, G. (1991), Attributive concept description with complement, *Artificial Intelligence* **48**, 1–26.

Schmolze, J. G. (1989), The language and semantics of NIKL, *Technical Report 89–4*, Department of Computer Science, Tufts University, Department of Computer Science, Tufts University, Medford, MA.

Suppes, P. (1976), Elimination of quantifiers in the semantics of natural language by use of extended relation algebra, *Revue Internationale de Philosophie* **30**, 243–259.

Suppes, P. (1979), Variable-free semantics for negations with prosodic variation, *in* E. Saarinen, R. Hilpinen, I. Niiniluoto and M. P. Hintikka (eds), *Essays in Honour of Jaakko Hintikka*, Reidel, Dordrecht, Holland, pp. 49–59.

Suppes, P. (1981), Direct inference in English, *Teaching Philosophy* **4**, 405–418.

Tarski, A. (1941), On the calculus of relations, *Journal of Symbolic Logic* **6**, 73–89.

Tarski, A. (1955), Contributions to the theory of models, Part III, *Indagationes Mathematicae* **17**, 56–64.

Wadge, W. W. (1975), A complete natural deduction system for the relational calculus, *Theory of Computation Report*, University of Warwick.

Woods, W. A. and Schmolze, J. G. (1992), The KL-ONE family, *Computers and Mathematics with Applications* **23**, 133–177.