University of Amsterdam
Programming Research Group

Division Safe Calculation in Totalised Fields

J.A. Bergstra
J.V. Tucker

Report PRG0605 September 2006

J.A. Bergstra

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
The Netherlands

tel. +31 20 525.7591
e-mail: janb@science.uva.nl

J.V. Tucker

Department of Computer Science
University of Wales Swansea

Singleton Park
Swansea, SA2 8PP
United Kingdom

e-mail: j.v.tucker@swansea.ac.uk

Programming Research Group Electronic Report Series

Division Safe Calculation in Totalised Fields[1]

J A Bergstra[2]

University of Amsterdam,
Informatics Institute,
Kruislaan 403,
1098 SJ Amsterdam,
The Netherlands

J V Tucker[3]

Department of Computer Science,
University of Wales Swansea,
Singleton Park,
Swansea, SA2 8PP,
United Kingdom

Abstract

A 0-totalised field is a field in which division is a total operation with $0^{-1} = 0$. Equational reasoning in such fields is greatly simplified but in deriving a term one still wishes to know whether or not the calculation has invoked 0^{-1}. If it has not then we call the derivation *division-safe*. We propose three methods of guaranteeing division-safe calculations in 0-totalised fields.

1 Introduction

The primary algebraic properties of the rational, real and complex numbers are captured by the operations and axioms of *fields*. The field axioms consist of the equations that define commutative rings and, in particular, two axioms, which are *not* equations, that define the inverse operator and the distinctness of the two constants. Traditionally, fields

[1]To refer to this paper cite as Research Report PRG 0605, Programming Research Group, University of Amsterdam, September 2006 or Technical Report CSR 14-2006, Department of Computer Science, University of Wales Swansea, September 2006.

[2]Email: janb@science.uva.nl

[3]Email: j.v.tucker@swansea.ac.uk

are *partial* algebras because the inverse operations are undefined at 0. The class of fields does *not* possess an equational axiomatisation.

However fields, especially the field of rational numbers and finite fields, are among the most important data types for computation. Rationals define measurements in the physical world and computer real arithmetic is based on a finite subset of the rational numbers. Computer integer arithmetic is based on finite rings and fields. All these fields are computable fields.

In [6, 7, 8], we have begun to investigate the field of rationals, and fields in general, using the elementary methods of abstract data type theory, especially equations, initial algebras and term rewriting. Calculations in fields are commonplace and the aim is to simplify algebraic reasoning and term rewriting for fields by removing the complications of partial functions and non-equational axioms.

A 0-*totalised field* is a field which has its inverse operator made total by imposing the equation

$$0^{-1} = 0.$$

If F is a field we denote the 0-totalised field by F_0, so the 0-totalised fields of rational $\mathbb{Q}$, real $\mathbb{R}$ and complex $\mathbb{C}$ numbers are denoted $\mathbb{Q}_0$, $\mathbb{R}_0$ and $\mathbb{C}_0$, respectively.

Interestingly, the study of 0-totalised fields leads to new axioms and structures. For example, an new equational theory called "elementary number algebra" (ENA) has been identified in [6] (there under the 'name' $CR{+}SIP{+}Ril$) as a single sorted finite equational specification for the operations $+, -, \cdot, ^{-1}$ which has all 0-totalised fields among its models and, in addition, a large class of commutative rings with inverses and 0-divisors. A model of ENA has been baptized a *meadow* in [6] and a theory of meadows is emerging.

Equational specification and reasoning in such totalised fields is indeed greatly simplified but in deriving a term one still wishes to know whether or not it has invoked 0^{-1}. Consider the calculation

$$\tfrac{1+1}{1+(-1)} + 1 = \tfrac{1+1}{0} + 1 = (1+1)\cdot 0^{-1} + 1 = (1+1)\cdot 0 + 1 = 1$$

in $\mathbb{Q}_0$ (or in any totalised field). Although the algebraic manipulation is simple we may wish to consider it *unsafe*, exceptional or, at least, special in some way: the calculation allows 0 in denominators and, moreover, makes use of the equation $0^{-1} = 0$. It is important to note that the outcome of the calculation is the valid term 1 and it is impossible to see from the outcome of the calculation that the derivation of the term involved unsafe steps.

The question to be discussed in this paper is this:

How do we detect and avoid unsafe divisions in calculations in 0-*totalised fields?*

If a calculation has not invoked 0^{-1} then we call it *division-safe*. We propose three methods of guaranteeing division-safe calculations in 0-totalised fields, as follows:

1. *Proof system:* Once a proof of $t = r$ has been found, prove additional information that implies that $t = r$ was derived in a division-safe way.

2. *Axioms:* Change the axioms of ENA to a weaker set that do not permit any division unsafe derivations.

3. *Algebra:* Modify a field to create a new algebra that satisfies all equations with division-safe proofs but fails to satisfy other equations.

Each of these methods has merit and works for fields in general. The key idea is for each term t to construct a new check term C_t such that

$$C_t = 1 \Longleftrightarrow \text{“}t \text{ can be evaluated in a division-safe way”}.$$

The origin of our work is found in two sources: a contemplation of recent work by Larry Moss and the objective to proceed with previous works on the algebraic specification of computable and semi-computable data types (in particular Bergstra and Tucker [1, 2, 3, 4]) in the context of data types relevant for the theory of computation over the real numbers.

Recently Moss found in [18] that there exists an equational specification of the ring of rationals (i.e., without division or inverse) with just *one* unary hidden function. He used a remarkable enumeration theorem for the rationals in Calkin and Wilf [9]. He also gave specifications of other rational arithmetics and asked if hidden functions were necessary.

In [6] we proved that there exists a finite equational specification under initial algebra semantics, *without* further hidden functions, but making use of an inverse operation, of the field of rational numbers. The existence of an equational specification using hidden functions follows from a result in [1], plus the observation that the rational number field is a computable algebra. The issue is to limit the use of hidden functions to useful and familiar operations. The fact that only a single hidden function is used depends upon special properties of the field of rational numbers. In [7] the specification found for the rational numbers was extended to the complex rationals with conjugation, and in [8] a specification was given of the algebra of rational functions with field and degree operations that are all total. In [**?**] we consider the situation for finite fields.

2 Elementary Algebraic Specifications (EAS)

2.1 Elementary algebraic specifications and totality

The theory of computable data types demonstrates that any computable system can be modelled using a finite set of equations or conditional equations under initial algebra semantics, possibly with the help of auxiliary or hidden functions.

In [7] we have discussed a very limited specification technique which we have termed *elementary algebraic specification* (EAS). The basic elements of EAS are as follows. We use algebraic specifications (Σ', E') of a *total* Σ algebra using a set E' of equations or conditional equations and initial algebra semantics such that $I(\Sigma', E')|_\Sigma \cong A$. In particular, the elementary specifications *require* total functions, *allow* hidden functions and sorts, and may or may not be complete term rewriting systems. Clearly, there are plenty of restrictions in force as there are many properties ruled out - see [7] for a long list with arguments for their omission. The *EAS specification problem* is this: Given a Σ algebra A, can one find an elementary algebraic specification (Σ', E') such that $I(\Sigma', E')|_\Sigma \cong A$.

An EAS is ‘better’ if it is finite rather than infinite, contains equations rather than conditional equations, or features nice term rewriting properties such as confluency and termination.

To use these EAS methods, we need to make algebras total that are usually considered to contain partial operators. Unavoidably, totalisation introduces an element of

arbitrariness or artificiality because values are added which are not based on the primary intuitions at hand.

Totalisation is not without problems when specifying a stack, as we have seen in our [5]. Totalisation is a matter of costs and benefits and in some cases the theory of a totalised data type, even when specified by means of a convincing EAS, may be harder to swallow than some of its non-elementary expositions, even including the required meta-theory for those non-elementary features. Stacks are a candidate of such a data type.

However, in the case of fields we have found totalisation and EAS methods convincing. For that we have four arguments:

(1) The EAS specification theory of totalised fields is rich and attractive.

(2) Totalisation of fields leads to a specification ENA which itself has a larger class of models, consisting of the so-called meadows and having remarkably natural properties.

(3) EAS provides a decoupling of syntax and semantics that is fundamental. All simple answers to the question why 0^{-1} fails to exist depend on the observation that this piece of syntax should not have been written down in the first place because it carries no intended meaning. Exactly this interplay between syntax and semantics is completely removed in the setting of EAS and totalised fields.

(4) The costs of totalisation, due to the introduction of a "fake" value for 0^{-1} and its impact on the theory of numbers are already compensated by the gains mentioned in (1) and (3) above.

2.2 Technical Preliminaries on Algebraic Specifications

We assume the reader is familiar with using equations and conditional equations and initial algebra semantics to specify data types. Some accounts of this are: ADJ [13], Meseguer and Goguen [11], or Wirsing [24].

The theory of algebraic specifications is based on theories of universal algebras (e.g., Wechler [23], Meinke and Tucker [17]), computable algebras (Stoltenberg-Hansen and Tucker [20]), and term rewriting (Terese [22]). The theory of computable fields is surveyed in Stoltenberg-Hansen and Tucker [21].

We use standard notations: typically, we let Σ be a many sorted signature and A a total Σ algebra. The class of all total Σ algebras is $Alg(\Sigma)$ and the class of all total Σ-algebras satisfying all the axioms in a theory T is $Alg(\Sigma, T)$. The word 'algebra' will mean total algebra.

3 Axioms for Number Algebras

The primary signature Σ is simply that of the *field*:

signature Σ
sorts $field$
operations
$0\colon \rightarrow field$;
$1\colon \rightarrow field$;

$+ : field \times field \to field$;
$- : field \to field$;
$\cdot : field \times field \to field$;
$^{-1} : field \to field$
end

3.1 Commutative Rings and Fields

The signature Σ_{CR} consists of Σ minus the inverse operator $^{-1}$. The first set of axioms is that of a *commutative ring with* 1, which establishes the standard properties of $+$, $-$, and $\cdot$.

equations CR

$$
\begin{aligned}
(x+y)+z &= x+(y+z) && (1)\\
x+y &= y+x && (2)\\
x+0 &= x && (3)\\
x+(-x) &= 0 && (4)\\
(x\cdot y)\cdot z &= x\cdot(y\cdot z) && (5)\\
x\cdot y &= y\cdot x && (6)\\
x\cdot 1 &= x && (7)\\
x\cdot(y+z) &= x\cdot y + x\cdot z && (8)
\end{aligned}
$$

end

These axioms generate a wealth of properties of $+, -, \cdot$ with which we will assume the reader is familar.

At this point there are different ways to proceed with the introduction of division. The orthodoxy is to add the following two axioms for fields: let *Gil* (general inverse law) denote the axiom

$$x \neq 0 \implies x \cdot x^{-1} = 1$$

and let *Sep* (the axiom of separation) denote

$$0 \neq 1.$$

Let (Σ, T_{field}) be the axiomatic specification of fields, where

$$T_{field} = CR + Gil + Sep.$$

3.2 Totalised Fields

In field theory the value of 0^{-1} is left undefined. However, in working with elementary specifications, operations are total. Thus, the class $Alg(\Sigma, T_{field})$ is the class of all possible *total* algebras satisfying the axioms in T_{field}; we refer to these algebras as *totalised fields*.

Now, for all totalised fields $A \in Alg(\Sigma, T_{field})$ and all $x \in A$, the inverse x^{-1} is defined. If 0_A is the zero element in A then, in particular, 0_A^{-1} is defined. The actual value 0_A^{-1} can be anything but it is convenient to set $0_A^{-1} = 0_A$ (see [6], and compare, e.g., Hodges [15], p. 695). A field A with $0_A^{-1} = 0_A$ is called *0-totalised*. This choice gives us a nice equational specification to use, the zero inverse law Zil:

$$0^{-1} = 0$$

With ZTF we specify zero totalised fields:

$$ZTF = CR + Gil + Sep + Zil.$$

Let $Alg(\Sigma, ZTF)$ be the class of all 0-totalised fields. One of the main Σ-algebras we are interested in is

$$\mathbb{Q}_0 = (\mathbb{Q}|0, 1, +, -, \cdot, ^{-1}) \in Alg(\Sigma, ZTF)$$

where the inverse is total $x^{-1} = 1/x$ if $x \neq 0$ and 0 if $x = 0$

Following [6] one may replace the axioms Gil and Sep by other axioms for division, especially, the three equations in an unit called SIP for *strong inverse properties*. They are considered "strong" because they are equations involving $^{-1}$ *without any guards*, such as $x \neq 0$:

equations SIP

$$(-x)^{-1} = -(x^{-1}) \tag{9}$$
$$(x \cdot y)^{-1} = x^{-1} \cdot y^{-1} \tag{10}$$
$$(x^{-1})^{-1} = x \tag{11}$$

end

In [6] we find that the following equations are provable:

Lemma 3.1. *$CR + SIP \vdash 0^{-1} = 0$ and $CR + SIP \vdash 0 \cdot x = 0$. Thus, $CR + SIP \vdash 0 \cdot 0^{-1} = 0$.*

In dealing with division it is helpful to have functions such as

$$Z(x) = 1 - x \cdot x^{-1} \text{ and } N(x) = x \cdot x^{-1}.$$

Clearly, $Z(x) = 1 - N(x)$ and $Z(x) = 0 \Leftrightarrow x \cdot x^{-1} = 1$.

In [6] (Theorem 3.5) an axiom L, based on Lagrange's Theorem, is used to give an equational specification of the the rationals. Lagrange's Theorem states that every natural number can be represented as the sum of four squares. We define a special equation L (for Lagrange):

$$Z(1 + x^2 + y^2 + z^2 + u^2) = 0.$$

L expresses that for a large collection of numbers, in particular those q which can be written as 1 plus the sum of four squares, $q \cdot q^{-1}$ equals 1.

Theorem 3.2. *There exists a finite elementary equational specification $(\Sigma, CR{+}SIP{+}L)$, without hidden functions, of $\mathbb{Q}_0$ under initial algebra semantics.*

3.3 ENA and Meadows

In [6] we also add to $CR + SIP$ the *restricted inverse law* (*Ril*):

$$x \cdot (x \cdot x^{-1}) = x$$

which, using commutativity and associativity, expresses that $x \cdot x^{-1}$ is 1 in the presence of x.

Definition 3.3. *We define the specification elementary number algebra* $ENA = CR + SIP + Ril$.

We note that:

Lemma 3.4. $Ril \vdash x \cdot x^{-1} = 0 \Longleftrightarrow x = 0$

For models of ENA the following convention is taken from [6]. A meadow satisfying *Sep* is called *non-trivial*. All total fields are clearly non-trivial meadows but not conversely. In particular, the prime fields $\mathbb{Z}_p$ of prime characteristic are meadows. That the initial algebra of $CR + SIP + Ril$ is not a field follows from the fact that $(1+1) \cdot (1+1)^{-1} = 1$ cannot be derivable because it fails to hold in the prime field $\mathbb{Z}_2$ of characteristic 2 which is a model of these equations as well.

Hirschfeld [14] has shown that equations SIP1 and SIP2 are derivable from SIP3 using $CR + Ril$.

4 Equational proof systems for safe division

4.1 Check terms and division safety

Let Σ be the signature of fields and $T(\Sigma, X)$ be the algebra of all Σ-terms with variables from X.

Definition 4.1. *To each term* $t(x_1, \ldots, x_n)$ *over* Σ *we assign a check term* $C_t(x_1, \ldots, x_n)$ *as follows:*

$C_0 = 1$
$C_1 = 1$
$C_x = 1 - 0 \cdot x$
$C_{t_1+t_2} = C_{t_1} \cdot C_{t_2}$
$C_{t_1 \cdot t_2} = C_{t_1} \cdot C_{t_2}$
$C_{t^{-1}} = C_t \cdot t \cdot t^{-1}$

Note that the term C_t has the same variables as t.

(In the definition above for totalised fields we could replace $C_x = 1 - 0 \cdot x$ with the simpler $C_x = 1$. We have used the more complicated term ready for the algebraic method later in Section 6; there we define *twin fields* in which the equation $0 \cdot x = 0$ is not valid in general.)

The idea of the construction of check terms is that:

$$C_t = 1 \Longleftrightarrow \text{inside-out evaluation of } t \text{ can be done in a division-safe way.}$$

For example, in a non-safe derivation containing 0^{-1}, we have

$$C_{0^{-1}} = C_0 \cdot 0 \cdot 0^{-1} = 1 \cdot 0 \cdot 0^{-1} = 0.$$

For instance,

$$C_{(x+y)/(z+1)} = C_{x+y} \cdot C_{1/z+1} = C_x \cdot C_y \cdot C_{z+1} \cdot (z+1)/(z+1) = \\ 1 \cdot 1 \cdot C_z \cdot C_1 \cdot (z+1)/(z+1) = (z+1)/(z+1).$$

Definition 4.2. *Let F_0 be a 0-totalised field. An equation $F_0 \models t = r$ valid in F_0 is said to be division safe if*

$$F_0 \models C_t = 1 \wedge C_r = 1.$$

The proof system method to ensure division safety is this: seek a set T of axioms for F_0, i.e., $F_0 \in Alg(\Sigma, T)$ such that each proof $T \vdash t = r$ can be complemented by proofs that $T \vdash C_t = 1$ and $T \vdash C_r = 1$.

4.2 Equational proof systems

Interestingly, we do not have far to look for one solution: consider initial algebra specifications. Suppose that there is a set E of equations such that $I(\Sigma, E) \cong F_0$. By initiality, equational reasoning is complete for closed identities relative to initial algebra specifications. Thus, for any *closed terms* t, r if $F_0 \models C_t = 1$ and $F_0 \models C_r = 1$ we have immediately $F_0 \vdash C_t = 1$ and $F_0 \vdash C_r = 1$.

Let us define this equational proof system method for division safety formally as follows:

Definition 4.3. *We write $F_0 \models_{ds} t = r$ in the 0-totalised field F_0 if*

$$F_0 \models t = r \text{ and } F_0 \models C_t = 1 \wedge C_r = 1.$$

We write $(\Sigma, T) \vdash_{ds} t = r$ if

$$(\Sigma, T) \vdash t = r \text{ and } (\Sigma, T) \vdash C_t = 1 \wedge C_r = 1.$$

The method works because we have:

Theorem 4.4. *Let F_0 be any totalised field and (Σ, E) any equational specification such that $I(\Sigma, E) \cong F_0$. Then for any closed terms t, r we have*

$$(\Sigma, E) \vdash_{ds} t = r \iff F_0 \models_{ds} t = r.$$

Proving $(\Sigma, E) \vdash_{ds} t = r$ is a general general approach to ensuring division safety; its practicality is dependent on the specification.

5 Equational axioms for weak safe division

5.1 Weak safe division in 0-totalised fields

We now consider a weaker notion of safety that has some interesting properties.

Definition 5.1. *Let F_0 be a 0-totalised field. An equation $F_0 \models t = r$ valid in F_0 is said to be weakly division safe if*

$$F_0 \models C_t = C_r.$$

Clearly, in a weakly division safe equation either both sides of the equation are safe or unsafe.

Compare the notion with divison safety (in Definition 4.2). There are equations that are weakly division safe but not necessarily division safe. For example, in any 0-totalised field F_0 we have $F_0 \models 0^{-1} = 0^{-1}$, which is weakly division-safe but is not division-safe.

Definition 5.2. *We write $F_0 \models_{wds} t = r$ in the 0-totalised field F_0 if*

$$F_0 \models t = r \text{ and } F_0 \models C_t = C_r.$$

We write $(\Sigma, T) \vdash_{wds} t = r$ if

$$(\Sigma, T) \vdash t = r \text{ and } (\Sigma, T) \vdash C_t = C_r.$$

For many equations $t = r$ where r is the simplified or "calculated" result or normal form of t it will be obvious by inspection that $F_0 \models C_r = 1$. In this case we have:

Lemma 5.3. *Suppose that $F_0 \models C_r = 1$. Then $\vdash_{wds} t = r$ implies $\vdash_{ds} t = r$.*

Lemma 5.4. *Let F_0 be a 0-totalised field and (Σ, E) be any specification true of F_0, i.e., $F_0 \models E$. Suppose every equation in E is weakly division safe for F_0. For every equation $t = r$ such that $(\Sigma, E) \vdash t = r$ then $t = r$ is weakly division safe.*

5.2 Meadows and the rationals

In the case of meadows and the rationals, we are able to weaken the axioms ENA and L we have used in such a way that

(i) all closed division-safe identities are provable; and

(ii) only weakly division-safe identities are provable.

In the light of Lemma 5.4, we start by checking the equations of our usual specification ENA. The following are the equations that are *not* weakly division safe.

(a) Additive Inverse: $x + (-x) = 0$ because it implies $0^{-1} + (-0^{-1}) = 0$.

(b) $(x^{-1})^{-1} = x$ because it implies $(0^{-1})^{-1} = 0$.

(c) $Ril : x \cdot x \cdot x^{-1} = x$ because it implies $0 \cdot 0 \cdot 0^{-1} = 0$.

It is possible to replace each of these equations in ENA by weakly division safe alternates as follows:

In the set CR of commutative rings axioms we replace additive inverse by these three equations

$$x + (-x) = 0 \cdot x, 0 \cdot 0 = 0, 0 \cdot 1 = 0$$

In the set SIP of inverse axioms the axiom above is replaced by:

$$(x^{-1})^{-1} = x \cdot x \cdot x^{-1}.$$

The axiom *Ril* is replaced by

$$x^{-1} \cdot x^{-1} \cdot x = x^{-1}.$$

Let ENA' be the new set of axioms. Then we have:

Lemma 5.5. *For any* 0*-totalised field* F_0 *we have* $F_0 \models ENA'$ *and since* ENA' *are weakly division-safe all the equational consequences of* ENA' *are division safe.*

Furthermore, in the special case of $\mathbb{Q}_0$ more can be shown. First, the Lagrange equation

$$L\colon\ Z(1 + x^2 + y^2 + z^2 + u^2) = 0$$

is not weakly divison-safe as may be seen on substituting 0^{-1} for the variables $x_1, \ldots, x_4$. But, the Lagrange axiom L can replaced by

$$Z(1 + x^2 + y^2 + z^2 + u^2) = 0 \cdot (x + y + z + u).$$

which is weakly division-safe.

Lemma 5.6. *For any closed terms* t, r

$$\mathbb{Q}_0 \models_{ds} t = r \text{ implies } ENA' + L' \vdash t = r$$

Proof. The proof is derived from the proof that $\mathbb{Q}_0 \cong I(\Sigma, ENA + L)$ from Bergstra and Tucker [6]. The proof of weak division safe identities between closed terms does not depend on non-division safe identities.

□

Thus, the axioms of $ENA' + L'$ is a reasonable specification of $\mathbb{Q}_0$ since it is a complete proof system for division-safe ground identities, and proves only weakly division-safe identities as well, though not all weakly division-safe identities.

6 Algebras for safe division

The third approach seeks a form of error algebra for fields, which are no longer 0-totalised fields. Then the idea is that ENA' and $ENA' + L'$ might be part of specifications for such algebras.

Given a field F of signature Σ we define a new Σ algebra F_{twin} such that

$$F_{twin} \models t = r \Longleftrightarrow F_0 \vdash_{wds} t = r$$

For each element $a \in F$ we make a copy $\hat{a} \in F_{twin}$ which represents the same value but in a division unsafe form. We may write $\hat{a} = a + 0^{-1}$. In a 0-totalised field we have $\hat{a} = a$, of course.

Twin fields are defined as follows. Let F be a field. Let F_0 be the 0-totalised form of F.

Definition 6.1. *The twin field extension of F is defined to be a Σ algebra with carrier $B \times F$; the constants $0, 1$ are*

$$(t, 0_F) \text{ and } (t, 1_F).$$

The operations are
$(b, x) +_{F_{twin}} (c, y) = (b \wedge c, x +_F y)$,
$(b, x) \cdot_{F_{twin}} (c, y) = (b \vee c, x \cdot_F y)$.
$(b, 0)^{-1} = (f, 0)$
$(b, x)^{-1} = (b, y)$ *where* $x \neq_F 0$ *and* $x \cdot y =_F 1$

Thus, F_{twin} contains an isomorphic copy of F, namely $\{t\} \times F$ and an isomorphic copy of F_0, namely $\{f\} \times F$. The inverse on the copy of F is made by: $(t, 0)^{-1} = (f, 0)$. Once an element lands in the error part of the twin field the operations keep it there. Notice that a twin field is not a field because

$$0 \cdot 0^{-1} \neq 0 \text{ and so } 0 \cdot x = 0 \text{ fails in } F_{twin}.$$

Lemma 6.2. *Let F be a field, F_0 be its 0-totalised form and F_{twin} its twin field. For any terms t, r, if $F_{twin} \models t = r$ then $F \models t = r$ and the equation is weakly division safe in F_0.*

Given this definition of F_{twin} we give a set of equations that can play a role similar to ENA:

$$ENA_{twin} = ENA' + \{0^{-1} \cdot x = 0^{-1}, (0^{-1} + x)^{-1} = 0^{-1} + x^{-1}, 0 \cdot x + 0^{-1} = 0^{-1}\}.$$

Using a proof similar to that of Theorem 3.2 in Bergstra and Tucker [6] we have:

Theorem 6.3. $\mathbb{Q}_{twin} \cong I(\Sigma, ENA_{twin} + L')$.

7 Concluding Remarks

Our work on the rationals and other fields can be viewed as a case study in abstract data types in which 'number algebra' is to be compared with 'process algebra' and other types of algebras that have been designed as elementary algebraic specifications to capture mechanisms found in the theory of computers and computation.

In this number algebra one takes the liberty to depart from the algebraist's orthodoxy (fields with their partial operations) and adapt the design of the algebras of numbers to the requirements of the computational modeling technique used, here elementary algebraic specifications (EAS). Thus, one can view this proposed topic 'number algebra' as a theory of arithmetics, including fields, shaped according to one of the many general modeling

techniques that have been developed in computer science: algebraic specifications where equational reasoning is extremely important. The topic is also an attempt to answer the question: What *can* one accomplish with the rationals and other fields *using simple equational reasoning only*?

Given its origins, the focus is on questions that one might pose from the computer science perspective: questions on specification, verification, prototyping, decidability and expressiveness. However, the theory of meadows is not without interest in pure algebra.

Assuming that one wants to view fields as total algebras, two strategies are feasible. First, use 0-totalised fields which possess nice equational specifications but which provide no protection against weak division unsafe conclusions. In this case protection against division unsafe results can be found via the use of additional proof obligations. An alternate is to use weaker equations.

Secondly, there are dedicated error algebras such as twin fields customised to the setting of fields. Each twin field contains a 0-totalised field as a substructure. Twin fields admit a specification theory similar to that of 0-totalised fields though require more complex equations. Twin fields guarantee that only weakly division safe conclusions are derived.

References

[1] J A Bergstra and J V Tucker, The completeness of the algebraic specification methods for data types, *Information and Control*, 54 (1982) 186-200.

[2] J A Bergstra and J V Tucker, Initial and final algebra semantics for data type specifications: two characterisation theorems, *SIAM Journal on Computing*, 12 (1983) 366-387.

[3] J A Bergstra and J V Tucker, Algebraic specifications of computable and semicomputable data types, *Theoretical Computer Science*, 50 (1987) 137-181.

[4] J A Bergstra and J V Tucker, Equational specifications, complete term rewriting systems, and computable and semicomputable algebras, *Journal of ACM*, 42 (1995) 1194-1230.

[5] J A Bergstra and J V Tucker, The data type variety of stack algebras, *Annals of Pure and Applied Logic*, 73 (1995) 11-36.

[6] J A Bergstra and J V Tucker, The rational numbers as an abstract data type, Research Report PRG0504, Programming Research Group, University of Amsterdam, August 2005 or Technical Report CSR12-2005, Department of Computer Science, University of Wales Swansea, August 2005. Submitted for publication.

[7] J A Bergstra and J V Tucker, Elementary algebraic specifications of the rational complex numbers, K Futatsugi et al, *Goguen Festschrift*, Springer Lecture Notes in Computer Science, vol. 4060, 459-475, Springer 2006.

[8] J A Bergstra, Elementary algebraic specifications of the rational function field, in A Beckmann et al, *Logical approaches to computational barriers. Proceedings of Computability*

in Europe 2006, Springer Lecture Notes in Computer Science, vol 3988, 40-54, Springer, 2006.

[9] N Calkin and H S Wilf, Recounting the rationals, *American Mathematical Monthly*, 107 (2000) 360-363.

[10] E Contejean, C Marche and L Rabehasaina, Rewrite systems for natural, integral, and rational arithmetic, in *Rewriting Techniques and Applications 1997*, Springer Lecture Notes in Computer Science vol. 1232, 98-112, Springer, Berlin,1997.

[11] J Meseguer and J A Goguen, Initiality, induction, and computability, In M Nivat (editors) *Algebraic methods in semantics*, Cambridge University Press,1986 pp 459 - 541

[12] J A Goguen, J W Thatcher, E G Wagner and J B Wright, Initial algebra semantics and continuous algebras, *Journal of ACM*, 24 (l977), 68-95.

[13] J A Goguen, J W Thatcher and E G Wagner, An initial algebra approach to the specification, correctness and implementation of abstract data types, in R.T Yeh (ed.) *Current trends in programming methodology. IV. Data structuring*, Prentice-Hall, Engelwood Cliffs, New Jersey, 1978, pp 80-149.

[14] Y Hirschfeld, Personal Communication, August 2006.

[15] W Hodges, *Model Theory*, Cambridge University Press, Cambridge, 1993.

[16] S Kamin, Some definitions for algebraic data type specifications, SIGLAN Notices 14 (3) (1979), 28.

[17] K Meinke and J V Tucker, Universal algebra, in S. Abramsky, D. Gabbay and T Maibaum (eds.) *Handbook of Logic in Computer Science. Volume I: Mathematical Structures*, Oxford University Press, 1992, pp.189-411.

[18] L Moss, Simple equational specifications of rational arithmetic, *Discrete Mathematics and Theoretical Computer Science*, 4 (2001) 291-300.

[19] L Moss, J Meseguer and J A Goguen, Final algebras, cosemicomputable algebras, and degrees of unsolvability, *Theoretical Computer Science*, 100 (1992) 267-302.

[20] V Stoltenberg-Hansen and J V Tucker, Effective algebras, in S Abramsky, D Gabbay and T Maibaum (eds.) *Handbook of Logic in Computer Science. Volume IV: Semantic Modelling* , Oxford University Press, 1995, pp.357-526.

[21] V Stoltenberg-Hansen and J V Tucker, Computable rings and fields, in E Griffor (ed.), *Handbook of Computability Theory*, Elsevier, 1999, pp.363-447.

[22] Terese, *Term Rewriting Systems*, Cambridge Tracts in Theoretical Computer Science 55, Cambridge University Press, Cambridge, 2003.

[23] W Wechler, *Universal algebra for computer scientists*, EATCS Monographs in Computer Science, Springer, 1992.

[24] M Wirsing, Algebraic specifications, in J van Leeuwen (ed.), *Handbook of Theoretical Computer Science. Volume B: Formal models and semantics*, North-Holland, 1990, pp 675-788.

Electronic Reports Series of the Programming Research Group

Within this series the following reports appeared.

[PRG0604] J.A. Bergstra and A. Ponse, *Projection Semantics for Rigid Loops*, Programming Research Group - University of Amsterdam, 2006.

[PRG0603] J.A. Bergstra and I. Bethke, *Predictable and Reliable Program Code: Virtual Machine-based Projection Semantics (submitted for inclusion in the Handbook of Network and Systems Administration)*, Programming Research Group - University of Amsterdam, 2006.

[PRG0602] J.A. Bergstra and A. Ponse, *Program Algebra with Repeat Instruction*, Programming Research Group - University of Amsterdam, 2006.

[PRG0601] J.A. Bergstra and A. Ponse, *Interface Groups for Analytic Execution Architectures*, Programming Research Group - University of Amsterdam, 2006.

[PRG0505] B. Diertens, *Software (Re-)Engineering with PSF*, Programming Research Group - University of Amsterdam, 2005.

[PRG0504] P.H. Rodenburg, *Piecewise Initial Algebra Semantics*, Programming Research Group - University of Amsterdam, 2005.

[PRG0503] T.D. Vu, *Metric Denotational Semantics for BPPA*, Programming Research Group - University of Amsterdam, 2005.

[PRG0502] J.A. Bergstra, I. Bethke, and A. Ponse, *Decision Problems for Pushdown Threads*, Programming Research Group - University of Amsterdam, 2005.

[PRG0501] J.A. Bergstra and A. Ponse, *A Bypass of Cohen's Impossibility Result*, Programming Research Group - University of Amsterdam, 2005.

[PRG0405] J.A. Bergstra and I. Bethke, *An Upper Bound for the Equational Specification of Finite State Services*, Programming Research Group - University of Amsterdam, 2004.

[PRG0404] J.A. Bergstra and C.A. Middelburg, *Thread Algebra for Strategic Interleaving*, Programming Research Group - University of Amsterdam, 2004.

[PRG0403] B. Diertens, *A Compiler-projection from PGLEc.MSPio to Parrot*, Programming Research Group - University of Amsterdam, 2004.

[PRG0402] J.A. Bergstra and I. Bethke, *Linear Projective Program Syntax*, Programming Research Group - University of Amsterdam, 2004.

[PRG0401] B. Diertens, *Molecular Scripting Primitives*, Programming Research Group - University of Amsterdam, 2004.

[PRG0302] B. Diertens, *A Toolset for PGA*, Programming Research Group - University of Amsterdam, 2003.

[PRG0301] J.A. Bergstra and P. Walters, *Projection Semantics for Multi-File Programs*, Programming Research Group - University of Amsterdam, 2003.

[PRG0201] I. Bethke and P. Walters, *Molecule-oriented Java Programs for Cyclic Sequences*, Programming Research Group - University of Amsterdam, 2002.

The above reports and more are available through the website: www.science.uva.nl/research/prog/

Electronic Report Series

Programming Research Group
Faculty of Science
University of Amsterdam

Kruislaan 403
1098 SJ Amsterdam
the Netherlands

www.science.uva.nl/research/prog/