Elementary Algebraic Specifications of the Rational Complex Numbers[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

From the range of techniques available for algebraic specifications we select a core set of features which we define to the *elementary algebraic specifications*. These include equational specifications with hidden functions and sorts and initial algebra semantics. We give an elementary equational specification of the field operations and conjugation operator on the rational complex numbers $\mathbb{Q}(i)$ and discuss some open problems.

For Joseph Goguen

[1] To refer to this paper cite as one of the following: Research Report PRG050?, Programming Research Group, University of Amsterdam, August 2005 or Technical Report CSR??-2005, Department of Computer Science, University of Wales Swansea, August 2005.

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

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

1 Introduction

Joseph Goguen has a vision for the theory of computation. It is algebraic, it is comprehensive, and it is focussed on the world's work. He uses a set of mathematical tools from category theory and universal algebra to explore a vast landscape of fundamental concepts, system architectures, emerging technologies, and contemporary practices in software development. He is a great explorer. His achievement is a fine example of just how much intellectual ground can be covered in the life time of a brilliant computer scientist with energy, curiosity, technical insight and a personal scientific agenda. He has reflected on his work on computing up to 1999 in Goguen [13]. There is so much to think about in this oeuvre.

One early line of thought is the role of initial algebras in semantic modelling and specification, expounded in [17]. Our own work in algebraic specifications from 1979 onwards owes a great debt to Joseph Goguen and his colleagues Jim Thatcher and Eric Wagner who, writing as the ADJ Group, provided a *perfect* mathematical basis for modelling and specifying abstract data types, starting in [18]. The ADJ Group established most of the basic theory by combining the technical ideas of many sorted algebras, equations, conditional equations, hidden functions and sorts, term rewriting and initial algebras. We continue to use many of their notations and techniques. Thanks to Sam Kamin [20], a rapidly growing literature was organised and problems identified and clearly stated, such as when were hidden functions and sorts necessary? Some of the more difficult problems of partiality, errors and parameterization also showed themselves in their early writing, problems which, after many long papers and books, are still not quite under control. Joseph Goguen and Eric Wagner have reflected on the ADJ Group in [11] and [35], respectively.

Our general theory of the algebraic specification of computable data types analysed the relationship between computability of abstract data types and equational specifications. Between 1979 and 1995 we published a series of papers that classified the computable, semicomputable and cosemicomputable data types using algebraic specifications (see, for example, Bergstra and Tucker [2, 3, 4, 5]). We proved several theorems that show that all computable data types have specifications that are very simple and small, or have good term rewriting properties, but always with hidden functions. Work has continued on this subject, refining notions such as finality (e.g., including Meseguer and Goguen [14] and Moss, Meseguer and Goguen, [29]), and on open questions (e.g., by Marongiu and Tulipani [23] and, most recently, by Khoussainov [21, 22]).

We have returned to the foundations of the subject in [7], tackling the specification of basic data types such as the rational numbers, and we continue here. First, we will select a core set of features which we define to be the *elementary algebraic specifications.* These are close to the basic techniques of the ADJ Group of the 1970s.

The set $\mathbb{Q}$ of rational numbers is a number system designed to denote measurements. They are used to define the real and complex numbers via approximation. The rationals are the numbers with which we make finite computations. Algebras made by equipping $\mathbb{Q}$ with some constants and operations we call *rational arithmetics.* We usually calculate with the algebra $(\mathbb{Q}|0, 1, +, -, \cdot, ^{-1})$ which is called the *field* of rational numbers when the operations satisfy certain standard axioms.

In addition to rational arithmetics, of particular interest are field extensions of the rational number field. Through Galois Theory, field extensions play a fundamental role in our understanding of the algebra of numbers, including the theory of equations ([10, 31]). One important field extension is the field of rational complex numbers, based on the set

$$\mathbb{Q}(i) = \{p + i \cdot q | p, q \in \mathbb{Q}\}.$$

This has special operations such as complex conjugation $cc(p + i \cdot q) = p - i \cdot q$.

The algebras of rational numbers, such as the field and its extensions by real and complex numbers, are among the truly fundamental data types. Despite the fact they have been known and used for over two millennia, they are neglected in the the modern theory of data types. After over 30 years of data type theory, many questions about rational arithmetics and their extensions are open.

Now the common rational arithmetics and field extensions are all computable algebras. Indeed, in the *theory of computable rings and fields* there is a wealth of constructions of computable algebras that start with the rationals and the finite fields: see the introduction and survey Stoltenberg-Hansen and Tucker [33]. Therefore, according to our general theory of algebraic specifications of computable data types they have various equational specifications under initial and final algebra semantics. Computable algebras even have equational specifications that are also complete term rewriting systems ([5]). However, these general specification theorems for computable data types involve hidden functions and are based on equationally definable enumerations of data.

Recently, in Moss [28], algebraic specifications of the rationals were considered. Among several interesting observations, Moss showed that there exists an equational specification of the ring of rationals (i.e., without division) with just *one* unary hidden function. He used a special enumeration technique based on a remarkable enumeration theorem for the rationals in Calkin and Wilf [8]. He also gave specifications of other rational arithmetics and asked if hidden functions were necessary. In [7] we proved that there exists a finite equational specification under initial algebra semantics, *without* hidden functions, of the field of rational numbers. The pursuit of this result leds to a thorough axiomatic examination of the divisibility operator, in which some interesting new axioms and models were discovered, and related results on fields and other rational arithmetics.

In particular, here we prove:

Theorem 1.1. *There exists a finite equational specification under initial algebra semantics, and without hidden functions, of the algebra*

$$(\mathbb{Q}(i) | 0, 1, i, +, -, \cdot, ^{-1}, cc)$$

of rational complex numbers with field and conjugate operations that are all total.

The structure of the paper is this. In Section 2, we discuss the basics of specification theory and define the elementary algebraic specifications. In Section 3, we describe the algebras and the axioms we will use to specify them. In Section 4 we prove the main theorem. Finally, in Section 5 we discuss some open problems.

This paper, and our [7], can be read independently but they are better viewed as a sequel to Bergstra and Tucker [4, 5], which contain many complementary results.

2 Elementary Algebraic Specifications

Since the first examples of algebraic specifications of data types in the 1970s, there has been a steady growth in the features that one may add to the basic techniques to be found the early ADJ papers such as [18]. The new techniques have been introduced for a number of obvious reasons: they have been found to be natural, or useful, or necessary to solve problems, or they have been used to extend or explore simpler techniques. The development of languages and tools (such as OBJ, ASF-SDF, Maude, CASL, etc.) for algebraic specification has increased the number and complexity of features in use.

So just what are the basic elements of this subject?

2.1 What are the elementary algebraic specifications?

Algebraic specification starts with the idea of modelling - e.g., data, processes, syntax, hardware, etc. - using sets and functions. Wherever there are sets and functions there are algebras! For example, the sets X, Y and function $f\colon X \to Y$ are combined to form the many sorted algebra $(X, Y | f)$. A particular algebra A is a mathematical model of a specific concrete representation of the system equipped with concrete operations. The need to understand the system, its representations and the extent to which they are unique leads to the concepts of (i) axiomatic theories for the chosen operators, and (ii) homomorphisms and isomorphisms for the comparison of algebras. The simplest axioms are equations. The simplest deductions are are those of equational logic based on the rewriting of terms. Any system can be modelled in this way. Therefore, we define the basic elements as follows.

Definition 2.1. *An algebraic specification (Σ', E') of a Σ algebra A is elementary if it involves only*

1. *A many sorted signature Σ' that is non-void. A signature is non-void if there is a closed term of every sort.*
2. *A set E' of equations or conditional equations.*
3. *An 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: see 2.2.

A standard way of proving an elementary specification is to check these properties:

Definition 2.2. *An algebraic specification (Σ', E') of a Σ algebra A satisfies Goguen's conditions if it the following are true:*

No Junk or Minimality *The algebra A is Σ-minimal.*

No Confusion or Completeness *For all closed Σ terms t, t', we have*

$$A \models t = t' \textit{ if, and only if, } E' \vdash t = t'.$$

In particular, the Goguen conditions imply that

$$I(\Sigma', E')|_{\Sigma} \cong A.$$

What makes these features *elementary*?

The purpose of developing a specification is to model, analyse and understand. In simple terms, these algebraic tools are fundamental for any modelling using sets and functions: they are used to abstract and analyse the properties models of an idea, component, or system. One chooses a set of operators and postulates a set of laws they satisfy; the laws are expressed as equations or conditional equations. The terms express all possible operations that can be derived by combining operations, and the equational identities express the consequent facts about the model. The term rewriting is a completely basic mechanism for both abstract reasoning and computation. This view suggests the elementary character of the equations and that we cannot make do with less. There is also an argument that they need extension in special circumstances.

Now, the whole modelling and specification process for elementary specifications is mathematically *robust* in the sense that the syntax and semantics have virtually no special conditions, neither subtle or obvious.

In modelling using an elementary algebraic specification one simply starts playing with operators, the equations and rewrites. There are no side conditions, side effects, and semantic errors to beware. The elementary algebraic specifications work simply in all cases. The only mistakes possible are mistakes in understanding what one is trying to model.

In our algebraic theory of computable data types, there are many results that show that if a many sorted algebra can be implemented on a computer then it possesses a range of *elementary equational specifications* with remarkable properties. Technically, all computable algebras can be specified with hidden functions, and all semicomputable algebras can be specified with hidden sorts and functions. In general this is the best possible. One theorem provides complete term rewriting systems ([5], Terese [34]). We need not worry about their power because:

The elementary algebraic specifications can specify everything that can be implemented on a computer in principle.

2.2 What are the non-elementary algebraic specifications?

What features have we excluded from the Definition 2.1 and hence have "declared" to be *not* elementary, and why?

We have excluded *final algebra semantics* because final algebras of equational specifications do not always exist and there are different interpretations possible (see Moss, Meseguer and Goguen [29]).

We have excluded *loose semantics* because we are focussed on specifying algebras up to isomorphism rather than classes of possible models.

We have excluded the following, too:

- **Generalisations of equations** One can use first order formulae that are "close" to equations such as Horn formulae. Since we exclude relations the Horn clauses are excluded. The *multi-equations* studied by Adamek et. al. [1] are also not simple enough.

- **Partial functions** Partiality is an essential aspect of computation. However, their logic is awfully complicated. Total functions are not without problems when specifying the stack, as we have seen in our [6]. However, as we showed in [7], it is not a problem to use total functions to specify the inverse on the rationals.

- **Errors and exceptions** The addition of new types of data, such as error flags, to familiar old friends, such as the natural numbers equipped with the predecessor $n - 1$, leads to difficult specifications and semantic complications.

- **Subsorts and order sorted algebras** Subsorts occur naturally and help with modelling subtyping, errors, etc. However, there are different theories none of which are simple: see, for example, Mosses on unified algebras [] or the survey [16].

- **Higher order** The higher-order theory is complicated from the start though it does possess a nice generalisation of the standard theory (see Meinke [27]).

- **Empty sorts** Empty sorts are tricky: see Goguen and Meseguer [15].

- **Priorities** Priorities for the equational rules are technically natural in developing software tools for algebraic specifications. However, they lead to complications since their term algebra representations do not satisfy the equations in general and must be considered pre-initial in some sense.

- **Modularity** Our elementary specifications are flat and do not have imports. Even the most simple notion of import introduces involved operations for flattening, see Rees et al [30].

- **Parameterization** There are many alternate treatments of parameterization, none of them simple.

Many of the features and techniques above that we have declared to be *not* elementary we certainly consider important. For example, features such as partiality and higher order equations are semantically fascinating and challenging to study, and are necessary to meet desires for certain kinds of specifications. However, they are not simple.

2.3 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 [18], Meseguer and Goguen [25], or Wirsing [37].

The theory of algebraic specifications is based on theories of universal algebras (e.g., Wechler [36], Meinke and Tucker [24]), computable and semicomputable algebras (Stoltenberg-Hansen and Tucker [32]), and term rewriting (Klop [26], Terese [34]).

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 Specifications for Rational Complex Numbers

3.1 Algebraic Specifications of the Rationals

We will build our specifications in stages. The primary signature Σ is simply that of the *field* of rational numbers:

signature Σ
sorts $field$
operations
$0\colon \ \to field$;
$1\colon \ \to field$;
$+\colon field \times field \to field$;
$-\colon field \to field$;
$\cdot\colon field \times field \to field$;
${}^{-1}\colon field \to field$
end

The first set of axioms is that of a *commutative ring with* 1, which establishes the standard properties of $+$, $-$, and $\cdot$. We will refer to these axioms by $CR1, \ldots, CR8$ etc.

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

Our first set SIP of axioms for ${}^{-1}$ contain the following, which we call the *strong inverse properties*. They are "strong" because they are equations in involving ${}^{-1}$ *without any guards*, such as $x \neq 0$:

equations SIP

$$\begin{aligned}
(-x)^{-1} &= -(x^{-1}) && (9)\\
(x \cdot y)^{-1} &= x^{-1} \cdot y^{-1} && (10)\\
(x^{-1})^{-1} &= x && (11)
\end{aligned}$$

end

Our specification $CR \cup SIP$ draws attention to division by zero:

Lemma 3.1. *The following equation is provable from $CR \cup SIP$:*

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

In particular, in our [7] (Theorem 3.5) we add a single axiom L to prove:

Theorem 3.2. *There exists a finite elementary equational specification $(\Sigma, CR \cup SIP \cup L)$, without hidden functions and under initial algebra semantics, of the rational numbers with field operations that are all total.*

In [7] we also add to $CR \cup SIP$ the *restricted inverse law* (*Ril*),

equations *Ril*

$$x \cdot (x \cdot x^{-1}) = x \qquad (12)$$

end

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

Whilst the initial algebra of CR is the ring of integers, we find that

Lemma 3.3. *The initial algebra of* $\mathrm{CR} + \mathrm{SIP} + \mathrm{Ril}$ *is a computable algebra but it is not an integral domain.*

The models of $CR + SIP + Ril$ are algebras with nice properties, in spite of not being fields nor even integral domains.

Definition 3.4. *A model of $CR + SIP + Ril$ is called a* meadow.

All fields are clearly meadows but not conversely (as the initial algebra is not a field).

Theorem 3.5. *For any closed terms $t, t' \in T(\Sigma)$, the following are equivalent*

1. *$t = t'$ is true in all totalised fields.*
2. *$t = t'$ is true in all totalised meadows.*

3.2 Algebraic Specifications of the Rational Complex Numbers

We add to the field signature Σ the complex conjugate operation $cc\colon \mathit{field} \to \mathit{field}$ to form the signature Σ_{cc}. Also to this signature we add the constant $i\colon\ \to \mathit{field}$ to form the signature $\Sigma_{cc,i}$. Consider these equations over the signature $\Sigma_{cc,i}$:

equations CC

$$\begin{aligned} cc(1) &= 1 && (13)\\ cc(\mathtt{i}) &= -\mathtt{i} && (14)\\ cc(x+y) &= cc(x)+cc(y) && (15)\\ cc(x\cdot y) &= cc(x)\cdot cc(y) && (16)\\ cc(-x) &= -cc(x) && (17)\\ cc(x^{-1}) &= (cc(x))^{-1} && (18)\\ cc(x\cdot x^{-1}) &= x\cdot x^{-1} && (19)\end{aligned}$$

end

3.3 Totalised Fields and Algebras satisfying the Specifications

The axioms of a field simply add to CR the following: the *general inverse law* (*Gil*)

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

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

$$0 \neq 1.$$

Thus, let $(\Sigma, T_{\mathit{field}})$ be the axiomatic specification of fields, where

$$T_{\mathit{field}} = CR \cup Gil \cup Sep.$$

The class $Alg(\Sigma, T_{\mathit{field}})$ is the class of *total* algebras satisfying the axioms in T_{field}. For emphasis, we refer to these algebras as *totalised fields*.

For all totalised fields $A \in Alg(\Sigma, T_{\mathit{field}})$ and all $x \in A$, the inverse x^{-1} is defined. In particular, 0_A^{-1} is defined. The actual value $0_A^{-1} = a$ can be anything but it is convenient to set $0^{-1} = 0$ (see [7], and compare, e.g., Hodges [19], p. 695). We use the specification $CR \cup SIP$ which forces $0^{-1} = 0$ (Lemma 3.1).

The main Σ-algebras we are interested in are these: first,

$$Q_0 = (\mathbb{Q} | 0, 1, +, -, \cdot, ^{-1})$$

where the inverse is total

$$\begin{aligned} x^{-1} &= 1/x && \text{if } x \neq 0;\\ &= 0 && \text{if } x = 0\end{aligned}$$

This total algebra satisfies the axioms of a field T_{field} and is a totalised field of rationals. Next, we are interested in the totalised field extension $Q_0(\mathtt{i})$ and its expansion by conjugation $Q_0(\mathtt{i}, cc)$.

4 Proof of Main Theorem

Theorem 4.1. *There exists a finite elementary equational specification* $(\Sigma_{cc,i}, E)$*, without hidden functions, of the algebra* $Q_0(cc, i)$ *of rational complex numbers with field and conjugate operations that are all total, under initial algebra semantics. That is,*

$$I(\Sigma_{cc,i}, E) = \mathbb{Q}_{cc,i}$$

Proof. Let (Σ, E) be any elementary equational specification without hidden functions of the totalised field of rationals $Q_0 = (\mathbb{Q}|0, 1, +, -, \cdot, ^{-1})$ so $I(\Sigma, E) = Q_0$. By Theorem 3.2, there is such an elementary specification. The strategy is to build a specification of $Q_0(cc, i)$ using real and imaginary parts, which are rationals.

Let Σ_{cc} be the field signature Σ extended by the complex conjugation operator

$$cc\colon \mathit{field} \to \mathit{field}.$$

First, we look at conjugation on $\mathbb{Q}$.

Conjugation on $\mathbb{Q}$ Conjugation on $\mathbb{Q} \subset \mathbb{C}$ is the identity function, $cc(r) = r$ for $r \in \mathbb{Q}$. Let $Q_0(cc)$ be the totalised field of rationals extended by conjugation cc. Let E^+ be the result of applying the following transformation of the equations in E: for each variable x in each equation of E substitute

$$\tfrac{1}{2}(x + cc(x)),$$

where $\frac{1}{2} = (1+1)^{-1}$. When applied to a complex number, the formula calculates its real part so when applied to a rational complex number from $Q_0(cc, i)$ it returns a rational number that would satisfy the equations of E.

Now define

$$E^+_{cc} = E^+ \cup \{cc(x) = x\} \cup \{\tfrac{1}{2}(x + x) = x\},$$

Lemma 4.2. $I(\Sigma_{cc}, E^+_{cc}) \cong Q_0(cc)$

Proof. We use Goguen's conditions.
No Junk The algebra $Q_0(cc)$ is clearly Σ_{cc} minimal since Q_0 is Σ minimal.
No Confusion By inspection, $Q_0(cc) \models E^+_{cc}$. We have to show completeness.

First, we make the observation that

$$E^+_{cc} \vdash E$$

since we can derive the equations of E by substituting back using the two axioms added as follows:

$$E^+_{cc} \vdash \tfrac{1}{2}(x + cc(x)) = \tfrac{1}{2}(x + x) = x.$$

Now suppose that $Q_0(cc) \models t_1 = t_2$ for any closed terms. By using axiom $cc(x) = x$ in E^+_{cc}, we can delete cc in the terms $t_1, t_2 \in T(\Sigma_{cc})$ to form $t'_1, t'_2 \in T(\Sigma)$ such that

$$E^+_{cc} \vdash t_1 = t'_1 \text{ and } E^+_{cc} \vdash t_2 = t'_2.$$

We know that

$$Q_0 \models t_1' = t_2'$$

and since is an initial algebra specification, we have

$$E \vdash t_1' = t_2'.$$

Hence, by the above observation,

$$E_{cc}^+ \vdash t_1' = t_2'.$$

and by applying $cc(x) = x$ as often as t_1', t_2' contain occurrences of cc

$$E_{cc}^+ \vdash t_1 = t_2.$$

This completes the argument. □

Let us replace the equation $cc(x) = x$ in E_{cc}^+ by the set

$$CCid = \{cc(t) = t | t \in T(\Sigma)\}$$

of all its closed Σ-term instances. We define:

$$E_{cc}^{++} = E^+ \cup CCid \cup \{\tfrac{1}{2}(x + x) = x\}.$$

Lemma 4.3. $I(\Sigma_{cc}, E_{cc}^{++}) \cong Q_0(cc)$

Proof. Replacing an equation by the set of all its closed instances does not change the initial algebra. In this case the cc's can be removed anyway. □

Now we consider the complex numbers

Conjugation on $\mathbb{Q}(i)$ Now consider the signature $\Sigma_{cc,i} = \Sigma_{cc} \cup \{i\colon \to field\}$, and the algebra $Q_0(cc, i)$ of rational complex numbers. We define the set

$$T = CR \cup SIP \cup Ril \cup CC \cup \{2 \cdot 2^{-1} = 1\}$$

Theorem 4.4. $I(\Sigma_{cc,i}, E^+ \cup T) \cong Q_0(cc, i)$

Proof. We verify the Goguen conditions.
No Junk Clearly, $Q_0(cc, i)$ is $\Sigma_{cc,i}$ minimal.
No Confusion By inspection,

$$Q_0(cc, i) \models E^+ \cup T.$$

For this we use the fact that the substitution of $\frac{1}{2}(x + cc(x))$ for each x in E guarantees that E is restricted to rational values which are the real parts of x when evaluated in $Q_0(cc, i)$.

To complete the argument we need some lemmas.

Lemma 4.5. *For each $t \in T(\Sigma_{cc})$, we have $T \vdash cc(t) = t$.*

Proof. This is an easy induction on t. □

Some useful consequences of this lemma are as follows. First, $T \vdash CCid$. Furthermore, we may deduce

$$T \vdash \tfrac{1}{2}(x + x) = \tfrac{1}{2} \cdot 2x = \tfrac{2}{2} \cdot x = 1 \cdot x = x$$

using CR and the axiom $2 \cdot 2^{-1} = 1$ in T. So we also have

$$E^{+} \cup T \vdash E_{cc}^{++}$$

Now suppose that $Q_0(cc, i) \models t_1 = t_2$ with $t_1, t_2 \in T(\Sigma_{cc,i})$. To show completeness we have to show that $E^{+} \cup T \vdash t_1 = t_2$.

Lemma 4.6. *For any closed term $t \in T(\Sigma_{cc,i})$ there are terms $p, q \in T(\Sigma)$ such that*

$$T \vdash t = p + i \cdot q.$$

Proof. We prove this by induction on terms.

Basis By the ring axioms of CR, the constants are as follows:

$$\begin{aligned} 0 &= 0 + i \cdot 0 \\ 1 &= 1 + i \cdot 0 \\ i &= 0 + i \cdot 1 \end{aligned}$$

Induction Step There are five cases, one for each operation. The cases of $+, -, \cdot$ are easy - here is one:

Let $t = t_1 \cdot t_2$ and suppose as induction hypothesis:

$$T \vdash t_1 = p_1 + i \cdot q_1 \text{ and } T \vdash t_2 = p_2 + i \cdot q_2.$$

Then, substituting, we calculate:

$$\begin{aligned} T &\vdash t = t_1 \cdot t_2 && \text{by assumption} \\ &\vdash t = (p_1 + i \cdot q_1) \cdot (p_2 + i \cdot q_2) && \text{by induction hypothesis} \\ &\vdash t = (p_1 \cdot p_2 - q_1 \cdot q_2) + i(q_1 \cdot p_2 + q_2 \cdot p_1) && \text{by axioms in } CR \end{aligned}$$

The other cases are more interesting.

Let $t = cc(t_0)$ and suppose as induction hypothesis:

$$T \vdash t_0 = p_0 + i \cdot q_0.$$

Then, substituting, we calculate:

$$\begin{aligned} T &\vdash t = cc(t_0) && \text{by assumption} \\ &\vdash t = cc(p_0 + i \cdot q_0) && \text{by induction hypothesis} \\ &\vdash t = cc(p_0) - i \cdot cc(q_0) && \text{by axioms of } CC \text{ in } T \\ &\vdash t = p_0 + i \cdot q_0 && \text{by Lemma 4.5} \end{aligned}$$

Let $t = r^{-1}$ and suppose as induction hypothesis:

$$T \vdash r = p + i \cdot q.$$

Then, substituting, we calculate:

$$\begin{array}{ll}
T \vdash t = r^{-1} & \text{by assumption} \\
\vdash t = \dfrac{1}{p + i \cdot q} & \text{by induction hypothesis} \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot \dfrac{1}{p + i \cdot q} \cdot (\dfrac{1}{p + i \cdot q})^{-1} & \text{by Ril in } T \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot \dfrac{1}{p + i \cdot q} \cdot (p + i \cdot q) & \text{by SIP} \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot \dfrac{p + i \cdot q}{p + i \cdot q} & \text{by axioms of CR} \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot cc(\dfrac{p + i \cdot q}{p + i \cdot q}) & \text{by axiom of } cc(x \cdot x^{-1}) = x \cdot x^{-1} \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot \dfrac{cc(p + i \cdot q)}{cc(p + i \cdot q)} & \text{by axiom of CC} \\
\vdash t = \dfrac{1}{p + i \cdot q} \cdot \dfrac{cc(p) - i \cdot cc(q)}{cc(p) - i \cdot cc(q)} & \text{by axioms of CC} \\
\vdash t = \dfrac{p - i \cdot q}{(p + i \cdot q) \cdot (p - i \cdot q)} & \text{by Lemma 4.5} \\
\vdash t = \dfrac{p}{p^2 + q^2} - i \cdot \dfrac{q}{p^2 + q^2} & \text{by axioms of T}
\end{array}$$

whch has the required form. □

Finally, to finish the completeness, suppose $Q_0(cc, i) \models t_1 = t_2$ with $t_1, t_2 \in T(\Sigma_{cc,i})$. By Lemma 4.6, we suppose that

$$T \vdash t_1 = p_1 + i \cdot q_1 \text{ and } T \vdash t_2 = p_2 + i \cdot q_2.$$

where $p_1, p_2, q_1, q_1 \in T()$. Hence,

$$Q_0 \models p_1 = p_2 \text{ and } Q_0 \models q_1 = q_2.$$

By Lemma 4.3, $I(\Sigma_{cc}, E^{++}_{cc}) \cong Q_0(cc)$ and so by completeness

$$E^{++}_{cc} \vdash p_1 = p_2 \text{ and } E^{++}_{cc} \vdash q_1 = q_2.$$

Next, thanks to Lemma 4.5, a consequence is

$$E^{+} \cup T \vdash E^{++}_{cc}.$$

Therefore,

$$E^{+} \cup T \vdash p_1 = p_2 \text{ and } E^{+} \cup T \vdash q_1 = q_2.$$

and so we are done with

$$E^{+} \cup T \vdash p_1 + i \cdot q_1 = p_2 + i \cdot q_2.$$

This completes the proof of Theorem 4.4. □

And hence the main theorem. □

5 Concluding remarks

There are open questions left over from the study of the rationals. For example, the following problem is quite basic:

Problem 5.1. *Is there a finite elementary equational specification of the totalised field Q_0, without hidden functions and under initial algebra semantics, which constitutes a complete term rewriting system?*

We know from our [5] that there exists such a specification with hidden functions.

However, questions proliferate as one reflects on the number of algebras using the rational numbers ([33]). For example, we do not know the answer to these simple questions.

Problem 5.2. *Is there a finite elementary equational specification of the field $Q_0(\boldsymbol{i})$ of rational complex numbers, without any hidden functions?*

Problem 5.3. *Is there a finite elementary equational specification of the algebra $Q_0(\boldsymbol{i}, cc)$, (without further hidden functions), which constitutes a complete term rewriting system?*

The rationals numbers constitute the data type for measurement with finite system of units and subunits. The real and complex numbers are constructed as completions of the rationals, using the idea of the approximation of measurements with unlimited accuracy. The real and complex numbers are the basis for vast range of data types used to model physical systems by means of measurement and equations (e.g., algebras of sequences, streams and signals, scalar and vector fields, continuous functions, probability distributions, and their abstractions). In general terms the data in these algebras are continuous data and they are built by some completion process from subalgebras containing discrete data, as the reals are made from the rationals.

Problem 5.4. *To create a comprehensive theory of computing, specifying and reasoning with systems based on continuous data. Ideally, the theory should integrate discrete and continuous data.*

At present this is a huge and complicated task as computation, specification and verification on continuous data are all active research areas. In fact, the task is a challenge in the special case of real numbers, see [7] for a discussion.

References

[1] J. Adamek, M. Hebert and J. Rosicky On abstract data types presented by multi-equations *Theoretical Computer Science* 275 (2002) 427 - 462

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

[3] 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.

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

[5] 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.

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

[7] J A Bergstra and J V Tucker, The rational numbers as an abstract data type, submitted.

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

[9] 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 1232, 98-112, Springer, Berlin,1997.

[10] H Edwards, *Galois theory*, Springer, l984.

[11] J A Goguen, Memories of ADJ, *Bulletin of the European Association for Theoretical Computer Science*, 36 (October l989), pp 96-102.

[12] J A Goguen, A categorical manifesto, *Mathematical Structures in Computer Science*, 1 (1991), pp 49-67.

[13] J A Goguen, Tossing algebraic flowers down the great divide, in C S Calude (ed.), *People and ideas in theoretical computer science*, Springer, Singapore, l999, pp 93-129.

[14] 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

[15] J Meseguer and J A Goguen, Remarks on remarks on many-sorted algebras with possibly emtpy carrier sets, *Bulletin of the EATCS*, 30 (1986) 66-73.

[16] J A Goguen and R Diaconescu An Oxford Survey of Order Sorted Algebra *Mathematical Structures in Computer Science* 4 (1994) 363-392.

[17] 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.

[18] 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, l978, pp 80-149.

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

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

[21] B Khoussainov, Randomness, computability, and algebraic specifications, *Annals of Pure and Applied Logic*, (1998) 1-15

[22] B Khoussainov, On algebraic specifications of abstract data types, in *Computer Science Logic: 17th International Workshop*, Lecture Notes in Computer Science, Volume 2803, 299 313, 2003

[23] G Marongiu and S Tulipani, On a conjecture of Bergstra and Tucker, *Theoretical Computer Science*, 67 (1989), .

[24] 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.

[25] J Meseguer and J A Goguen, Initiality, induction and computability, in M Nivat and J Reynolds (eds.), *Algebraic methods in semantics*, Cambridge University Press, Cambridge, 1985, pp.459-541.

[26] J W Klop, Term rewriting systems, in S. Abramsky, D. Gabbay and T Maibaum (eds.) *Handbook of Logic in Computer Science. Volume 2: Mathematical Structures*, Oxford University Press, 1992, pp.1-116.

[27] K Meinke, Universal algebra in higher types, *Theoretical Computer Science*, 100 (1992) ??-??.

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

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

[30] D Rees, K Stephenson and J V Tucker, The algebraic structure of interfaces, *Science of Computer Programming*, 49 (2003), 47-88.

[31] I Stewart, *Galois theory*, Chapman and Hall, l973.

[32] 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.

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

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

[35] E Wagner, Algebraic specifications: some old history and new thoughts, *Nordic Journal of Computing*, 9 (2002), 373 - 404.

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

[37] 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.