Electronic Notes in Theoretical Computer Science 42 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume42.html 16 pages

Using Theory Interpretation to Mechanise the Reals in a Theorem Prover

Jamie Shield [1]

Software Verification Research Centre
The University of Queensland, Australia 4072

Ian Hayes [2] and David Carrington [3]

School of Computer Science and Electrical Engineering
The University of Queensland, Australia 4072

Abstract

The mechanisation of the real numbers within theorem provers is of practical benefit for the verification of real-time systems. The real numbers provide a foundation within the theorem prover for classical mathematical analysis such as differentiation and integration. The approach we have taken makes extensive use of the theory interpretation facilities of the interactive theorem prover Ergo to maximise theory reuse and hence minimise theorem redundancy. The theory developed is compared with Harrison's HOL version.

1 Introduction

This paper describes the development and subsequent analysis of supporting real numbers using theory instantiation and interpretation within the Ergo theorem prover [5]. Although Ergo was chosen because of our familiarity with it, the work reported here should be applicable to any theorem prover that provides facilities for theory instantiation and interpretation. Ergo can be customised to support any logic, and supports the *window inference* proof paradigm [6]. Window inference was designed to support hierarchical, goal-directed proofs and to provide support for the context of a subterm. Ergo uses Qu-Prolog [7] as a meta-language for high-level tactics. It provides significant support for recording, browsing and replaying proofs, has access to a directed

[1] Email: jims@csee.uq.edu.au
[2] Email: ianh@csee.uq.edu.au
[3] Email: davec@csee.uq.edu.au

acyclic graph of theories and a powerful query language with which to search the theories.

The motivation for this project stems from a larger project that aims to provide tool support for a real-time refinement calculus [4,9]. The real numbers are continuous and it is this property that makes them desirable (if not necessary) when dealing with real-time systems. For example, in a system involving a temperature sensor, to specify limits on the rate of change of temperature standard differential calculus (based on the reals) may be used. If the reals are not available, only weaker properties of a discrete model can be stated formally.

2 Representing the Reals

There are two approaches for constructing the reals, axiomatisation and definitional extension. Axiomatisation of the reals involves the declaration of a new type, operators that use the new type, and the axioms of the reals. The alternative approach, definitional extension, involves defining the reals in terms of existing constructs. This involves initially creating a model of the reals. For instance, the reals could be modelled by Cauchy sequences (converging sequences of rationals). The standard axioms of the reals can be proved as theorems of the model.

Mathematicians have developed a rich variety of algebraic structures such as monoids, groups, rings and fields. The algebraic structures were identified by mathematicians via the axioms defining them, and the theorems developed are properties shared amongst many mathematical structures. For instance, the reals share many properties with rational, integer and natural numbers. Once a model has been shown to maintain the axioms of an algebraic structure, the theorems of the algebraic structure can be used to reason about the model. For example, the integers form an abelian group with addition as the operator, zero as the identity and negation as the inverse. The benefits of introducing the more general algebraic structures include increased reuse of the theories and increased confidence in the validity of the system.

As a consequence of many previous constructions of the reals, three techniques for modeling the reals have emerged [3].

Positional expansions The most intuitive method for representing the reals is by positional expansions. These are infinite sequences of numbers (positional sequences) of any base, e.g., decimal or binary. Elegant arithmetic is not supported and formal analysis is tedious when this method is used [3, pp.13-16].

Dedekind cuts Dedekind [1] represents a real number by the set of all rational numbers less than it. This set is referred to as a cut. Cuts must satisfy several properties. Namely, the cut may not be empty; it must not be the set of all rationals, $\mathbb{Q}$; it must be downwards closed, that is, given

any element in the cut, all rationals less than that element must also be in the cut. Finally, there must exist no greatest element in the cut, that is, for every element in the cut there is another, larger element, also in the cut.

The construction of the arithmetic operations is quite straightforward. For instance, for cuts (sets of rationals) X and Y,

$$X + Y = \{x + y \mid x \in X \wedge y \in Y\}$$

The definition of real multiplication does complicate proofs, however. Given that a cut contains all rationals less than the real it is representing, some of those elements are negative. The resulting set, naively using the following incorrect definition of real multiplication, contains arbitrarily large positives as the product of two negative rationals is positive. Consequently, the resulting set is not a cut.

$$X * Y = \{x * y \mid x \in X \wedge y \in Y\}$$

Cauchy sequences This method represents a real number by the set of rational sequences that converge to it. Harrison [3] observes that the Cauchy sequences method which is usually attributed to Cantor was also explored by Meray.

The approach taken in our work was based on Cauchy sequences. They are simpler to handle than positional expansions, and more familiar than Dedekind cuts. The structure of the paper is as follows: the theory of interpretation and instantiation are reviewed in Section 3. Implementation details are provided in Section 4. In Section 5, this project is compared with the mechanisation of the Cauchy sequences approach within HOL [3]. Section 6 summarises the outcomes of the project. Summaries of every theory in the development may be found in Appendix A of [8].

3 Interpretation and Instantiation

The traditional development of the real numbers has produced a well-defined hierarchy of algebraic structures. This hierarchy of monoids, fields, etc. is quite modular and provides opportunity for theory reuse. Ergo provides extensive support mechanisms for theory reuse via the following mechanisms:

Instantiation Theory instantiation is used for syntactic inclusion. When a theory is instantiated, it is as though the instantiation command is textually replaced by the theory being instantiated, except that the theory symbols are renamed. The theory properties are copied, under the renamings. Note that axioms from the theory being instantiated remain as axioms in the instantiated theory. For example, given a theory of semigroups, and a theory of identities, a theory of monoids could be created by instantiating them both into a new theory.

Interpretation Theory interpretation is used when the theorems of a theory M, are desired for use in another theory N. The theorems of M are valid in N provided the axioms of M can be shown to hold using the properties of N. Like instantiation, Ergo interpretation copies the properties of a theory into another, under the renamings, except axioms are mapped to postulates, which must be proven in the new context.

Theory interpretation can be used if a theory (e.g., the integers under addition) is believed to be a model of, or possess the properties of, another theory (e.g., a group).

More formally, a theory presentation is a pair $P = < L, G >$ consisting of a first order language L and the set of sentences G in L. A translation $I : L_1 \rightarrow L_2$ is a mapping from one language L_1 to another L_2 that renames each symbol of L_1 to a symbol of L_2 with the same syntactic declarations (e.g., arity). Given $P_1 = < L_1, G_1 >$ and $P_2 = < L_2, G_2 >$, $I : L_1 \rightarrow L_2$ is an interpretation if $I(\sigma) \in G_2$ whenever $\sigma \in G_1$ [2].

Hierarchies of theories

More complex theories can be built by combining theories via instantiation and/or interpretation. As an example, a ring can be axiomatised by combining the axiomatisations of an abelian group (instantiating the operator of the group with the '+' operator of the ring) and a semigroup (instantiating the operator of the semigroup with the '$*$' operator of the ring), and the abelian group theory can itself be axiomatised in terms of (another) instantiation of the semigroup theory. The theory for the ring thus contains two instantiations of the semigroup theory, but as the names of the operators and their identity elements are distinct, no confusion arises.

To show that the integers $\mathbb{Z}$ with operators '+' and '$*$' form a ring, the ring theory is interpreted into the theory of integers with appropriate renaming of the ring's abstract operators. For the interpretation to be valid, the ring axioms must be proven as theorems using the properties of the integers, integer addition and integer multiplication. Once the ring axioms have been proven to hold for the integers, the ring theorems can be used to reason about the integers. Hamilton et al. present a more thorough discussion of interpretation in Ergo [2].

The practical benefits of theory interpretation and instantiation are:

Theory reuse New theories can be built from existing (in some cases well known) theories, rather than being axiomatised afresh. This reduces both the work required and the likelihood of introducing errors in the axiomatisation. For example, in an imperative programming language, commands with sequential composition as the operator and the null command as the identity, form a monoid.

By introducing theories to structure the development of a new theory, one provides new theories that could be of use in future applications. The more general the theory, the more likely it is to be reused. In practice, a

general theory may not be recognised until the similarity between two sets of axioms used in different contexts is recognised. In this case it may be beneficial to extract the similar axioms to form a new theory and instantiate it into the separate contexts.

Theorem reuse All (abstract) theorems proved for a theory are immediately available in an instantiation of that theory, and also for an interpretation of the theory, provided in the latter case that the axioms of the interpreted theorem have been shown to hold. This provides a powerful mechanism for modularising proof, that has been successfully exploited by mathematicians over the years.

4 Design and Implementation

4.1 Framework

Our first step in the development of a theory for the reals $\mathbb{R}$ was the axiomatisation of the field and ordered field algebraic structures. The hierarchy of supported theories is shown in Figure 1. The figure uses the term inheritance. Inheritance is a trivial form of instantiation in which no renaming takes place. The axiomatisation of the ordered fields in this bottom-up development began with the **semigroup**, **commutativity**, **inverse** and **identity** theories. This was followed by the **monoid** theory, then **group**, **abeliangroup** etc. up to the theory of ordered fields. This theory hierarchy increases the opportunities for theory reuse while still constraining the amount of theory development required.

There are several approaches for building the axiomatisation of algebraic structures up to and including the ordered field theory. While theory interpretation/instantiation was advocated in Section 3, other alternatives need to be considered to ensure the best decision is made. The simplest method for axiomatising ordered fields is to create a single theory (with no subtheories) containing all the ordered field axioms, definitions and theorems. The lack of modularity, lack of theorem traceability, and repetition of many similar theorems are concerns with this approach. A slightly more sophisticated approach is to create a theory for each algebraic structure and to produce a linear chain of inherited theories. This approach solves the problem of modularity but duplicates theories. The chosen alternative for axiomatising ordered fields involves the use of theory interpretation and instantiation. With theory interpretation, fifteen theories replace an estimated forty-eight theories (fourteen for field properties of ordered pairs to construct rationals, seventeen for ordered field properties of rationals and seventeen for ordered field properties of the reals). The particular numbers are not important. Much theory development, however, has been avoided and using theory interpretation has increased the opportunities for theory and theorem reuse.

After axiomatisation of the ordered field theory, models are created of

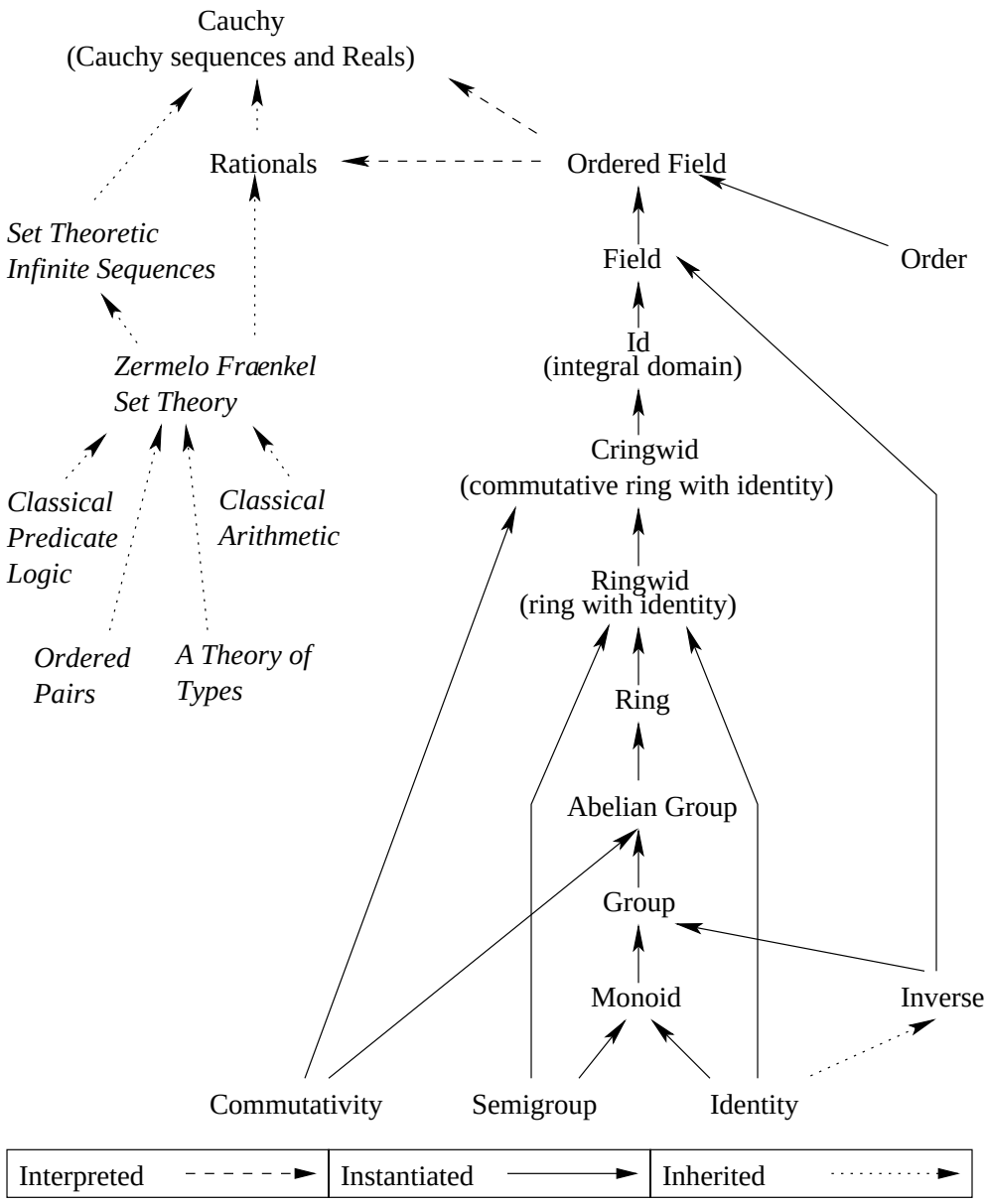

Fig. 1. Supported Theories

the rationals and subsequently the reals. Several standard Ergo theories are directly used for the creation of the models. For the rationals, the Zermelo Frænkel Set Theory (**zfc**) is inherited. This has several subtheories. Directly used for the construction of the model of the rationals are the ordered-pairs theory (**o_pair**), Classical arithmetic theory (**carith**): which is a theory of integers based on Peano arithmetic, Classical predicate logic (or First Order Logic: FOL) theory (**cpred**) and types theory (**types**). For the construction of the model of the reals, a standard theory of set theoretic infinite sequences is used (**seq**). This theory inherits the **zfc** theory.

The rationals are modelled as equivalence classes of ordered pairs, $\mathbb{S}$, in which the second element of each pair is non-zero.

Definition 4.1 (Ordered Pairs) *The definition of the set of ordered pairs is:*

$$\mathbb{S} \mathrel{\widehat{=}} \mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$$

These ordered pairs do not give a unique representation for a rational number. Hence we introduce the equivalence relation ρ_{s}, and the equivalence class corresponding to an element of $\mathbb{S}$.

Definition 4.2 (ρ_{s}) *The equivalence relation, ρ_{s}, on $\mathbb{S}$ is defined as:*

$$\frac{(a, b) \in \mathbb{S} \land (c, d) \in \mathbb{S}}{((a, b)\ \rho_{\mathrm{s}}\ (c, d)) \equiv (a * d = c * b)}$$

Definition 4.3 (*equivclass*) *Given A, an element of $\mathbb{S}$, A's equivalence class is defined as:*

$$[A]_{\rho_{\mathrm{s}}} \mathrel{\widehat{=}} \{w : \mathbb{S} \mid (A\ \rho_{\mathrm{s}}\ w)\}$$

Definition 4.4 (*rationals_set*) *The rationals are the set of all equivalence classes, as determined by ρ_{s}.*

$$\mathbb{Q} \mathrel{\widehat{=}} \{w : \mathbb{P}(\mathbb{S}) \mid \exists\, z : \mathbb{S} \bullet w = [z]_{\rho_{\mathrm{s}}}\}$$

Although a model of the rationals has been defined, the typical axioms and theorems are not yet available. However, by interpreting the ordered field theory into the rationals theory, using $\mathbb{Q}$ as the set, and with suitable definitions of the field operators on elements of this set, all that is required is to show that the axioms of an ordered field hold for the rationals, and then all theorems of ordered fields are available to reason about the rationals.

Next, the reals are modelled using the rationals and infinite sequences. Infinite Cauchy sequences are defined in terms of sequences of rationals that converge. Finite Cauchy sequences, when required, are modelled by recurring sequences, i.e., by repeating the last rational ad infinitum.

Definition 4.5 (*rationals_seq*) *The set of rational sequences is defined as:*

$$\mathrm{seq}(\mathbb{Q}) \mathrel{\widehat{=}} \mathbb{N} \to \mathbb{Q}$$

Definition 4.6 (*cauchy_seq*) *Given the positive rationals, $\mathbb{Q}^+$, the rationals' ordering operation, $<_{\mathbb{Q}}$, the rationals' subtraction operation, $-_{\mathbb{Q}}$, and the rationals' absolute value operation, $|\cdot|_{\mathbb{Q}}$, the Cauchy sequences are defined as:*

$$\mathbb{Q}^{\omega} \mathrel{\widehat{=}} \{v : \mathrm{seq}(\mathbb{Q}) \mid \forall\, e : \mathbb{Q}^+ \bullet \exists\, ne : \mathbb{N} \bullet \\ \forall\, m, n : \mathbb{N} \bullet m > ne \land n > ne \Rightarrow |v[m] -_{\mathbb{Q}} v[n]|_{\mathbb{Q}}\ <_{\mathbb{Q}} e\}$$

$\mathbb{Q}^\omega$ is the set of all rational sequences that for all positive rationals e, there is some index in the sequence after which all pairs of elements are closer than e.

Definition 4.7 (*zero_seq*) *The rational sequences in $0_{\mathbb{Q}^\omega}$, which is the model of zero in the reals, are also Cauchy sequences [1].*

$$0_{\mathbb{Q}^\omega} \mathrel{\widehat{=}} \{v : seq(\mathbb{Q}) \mid \forall\, e : \mathbb{Q}^+ \bullet \exists\, ne : \mathbb{N} \bullet \\ \forall\, n : \mathbb{N} \bullet n > ne \Rightarrow |v[n]|_{\mathbb{Q}} <_{\mathbb{Q}} e\}$$

Definition 4.8 ($\rho_{\mathbb{Q}^\omega}$) *The $\rho_{\mathbb{Q}^\omega}$ operator is the equivalence relation for Cauchy sequences. Two Cauchy sequences are equivalent if, when subtracted, where $-_{\mathbb{Q}^\omega}$ is the (pointwise) subtraction operator for the Cauchy sequences, the result is in $0_{\mathbb{Q}^\omega}$.*

$$\rho_{\mathbb{Q}^\omega}(X, Y) \mathrel{\widehat{=}} (X -_{\mathbb{Q}^\omega} Y) \in 0_{\mathbb{Q}^\omega}$$

Definition 4.9 (*cequivclass*) *The definition of the equivalence classes on the rationals is a trivial use of the $\rho_{\mathbb{Q}^\omega}$ operator.*

$$[A]_{\rho_{\mathbb{Q}^\omega}} \mathrel{\widehat{=}} \{v : \mathbb{Q}^\omega \mid A \ \rho_{\mathbb{Q}^\omega}\ v\}$$

Definition 4.10 (*reals*) *The reals are defined as equivalence classes of Cauchy sequences.*

$$\mathbb{R} \mathrel{\widehat{=}} \{v : \mathbb{P}(\mathbb{Q}^\omega) \mid \exists\, n : \mathbb{Q}^\omega \bullet v = [n]_{\rho_{\mathbb{Q}^\omega}}\}$$

Finally, the reals are shown to form a complete ordered field. This is achieved by proving the ordered field axioms hold as well as the supremum property. The supremum property is that, for any non-empty subset, x, of the reals, if x has an upper bound (m) in $\mathbb{R}$ then it has a least upper bound in $\mathbb{R}$.

$$\forall\, x : \mathbb{P}(\mathbb{R}) \bullet x \neq \varnothing \wedge (\exists\, m : \mathbb{R} \bullet UB(m, x)) \Rightarrow (\exists\, m : \mathbb{R} \bullet LUB(m, x))$$

where $UB(m, x)$ indicates that m is an upper bound of x, and $LUB(m, x)$ indicates that m is the least upper bound of x.

$$UB(m, x) \mathrel{\widehat{=}} \forall\, y : x \bullet y \leq m$$
$$LUB(m, x) \mathrel{\widehat{=}} UB(m, x) \wedge \forall\, v : \mathbb{R} \bullet UB(v, x) \Rightarrow m \leq v$$

4.2 Implementation in Ergo

The system is built in a bottom-up fashion starting with axiomatisation of the **semigroup**, **identity**, **inverse** and **commutative** theories. Next, the **semigroup** properties and the **identity** properties are instantiated into the **monoid** theory. The Ergo [5] source of the **identity** theory is:

```
% First the identity theory is created.
  add_theory(identity).
```

```
% Next various theories (and "all" their subtheories)
% are inherited, e.g., the types
% theory allows reasoning about types.
  include_theory([types,undef],all).
% The operator and the set are declared.
  declare 'op'(_,_).
  declare set.
% The add_operator command defines the precedence and
% associativity of infix operators.
  add_operator(500,leftassoc,'op').
% Next, the identity element is defined.
  declare id.
% Various axioms about the theory symbols are provided.
  axiom id_in_set ===  id:set.
  axiom identity1 ===  X:set => id op X = X.
  axiom identity2 ===  X:set => X op id = X.
% Finally the theory is closed.  This disallows any
% updates to the axioms of the theory.
% Theorem development is still possible.
  close_theory.
```

The axiomatisation of **semigroup** theory is listed below.

```
% The semigroup theory is created.
  add_theory(semigroup).
% Other theories are inherited.
  include_theory([types,undef],all).
% The operator and the set are declared.
  declare 'op'(_,_).
  declare set.
  add_operator(500,leftassoc,'op').
% The closure and associativity axioms are provided.
  axiom closure ===
   X:set and Y:set =>
    (X op Y):set.
  axiom assoc ===
    X:set and Y:set and Z:set =>
     (X op Y) op Z = X op (Y op Z).
  close_theory.
```

A monoid is a semigroup that contains an identity. Consequently, both the **semigroup** and the **identity** theories need to be instantiated. The Ergo source of the **monoid** theory is shown below.

```
% Create the monoid theory and inherit other theories.
  add_theory(monoid).
  include_theory([types,undef],all).
```

```
% Declare the monoid's set, operator and identity.
  declare set.
  declare 'op'(_,_).
  add_operator(500,leftassoc,'op').
  declare id.
% Next, the semigroup theory is instantiated into the
% subtheory monoid_sg.
  instantiate(semigroup,monoid_sg,
            ['set' ---> set,
             T 'op' R ---> T op R],
            [],
            []).
% Finally, the identity theory is instantiated into the
% subtheory monoid_id.
  instantiate(identity,monoid_id,
           ['set' ---> set,
            T 'op' R ---> T op R,
            'id' ---> id],
           [],
           []).
  close_theory.
```

The first instantiation command in the **monoid** theory copies the closure and associativity axioms from the **semigroup** theory into the **monoid** theory. The first argument of the command is the more general theory from which specialised theorems are created. The second argument is the name given to the subtheory of the current theory into which the axioms and theorems of the **semigroup** theory are copied. That is, the instantiation creates a subtheory of the current theory that contains the remapped axioms and theorems. The third argument contains the operator and constant renamings that are applied to transform the axioms and theorems of the general theory into the more specialised axioms and theorems. The fourth argument maps ancestor theories from the general theory to an existing interpreted ancestor of the specialised or current theory. This paper does not require the use of this argument. The fifth argument remaps theories interpreted within the general theory to subtheories of the current theory. Although this mapping is empty for the above examples, it is used below for the interpretation of the **ordered field** theory into the **rationals**.

The process of axiomatisation, provision of theorems and instantiation continues until the **ordered field** theory has been developed. At this stage, the rationals are modelled and the **ordered field** theory is interpreted into the **rational** theory to obtain the **ordered field** properties. All axioms of the **ordered field** theory must be proved within the context of the **rational** model before the theorems from the **ordered field** are valid for rationals.

Given Ergo's integer theory, the rationals are modelled as equivalence classes of ordered pairs. The ordered pairs used in the model are interpreted with the **field** properties and the rationals are interpreted with the **ordered field** properties. That is, given the definition of the type of ordered pairs of integers, `s`, and the equivalence relation, `s=` (which is ρ_s from Definition 4.2), equivalence classes are defined.

```
define equivclass(A) ===
       set_of w:s (A s= w).
```

Next, given Ergo's existential quantifier, `ex`, and the powerset operator, `power`, the model of the rationals is formed.

```
define rationals_set ===
    set_of w:power(s) (ex z:s (w=equivclass(z))).
```

The rationals are an ordered field. Hence we interpret from the ordered field theory into the theory of the rationals and show that the axioms of ordered fields hold for the rationals. The operators and constants of the rationals are prefixed with 'q' to avoid name clashes.

```
interpret(ofield,rationals_ofield,
          ['set'        ---> rationals_set,
           'pos_set'    ---> pos_rationals,
           'neg_set'    ---> neg_rationals,
           T '+' R      ---> T 'q+' R,
           T '*' R      ---> T 'q*' R,
           'zero'       ---> 'qzero',
           'one'        ---> 'qone',
           '-'(S)       ---> 'q-'(S),
           'inverse'(S) ---> 'qinverse'(S),
           'abs'(T)     ---> 'qabs'(T),
           T '<' R      ---> T 'q<' R,
           T '=<' R     ---> T 'q=<' R],
          [],
          [ofield_sg1           ---> rationals_sg1,
           ofield_sg2           ---> rationals_sg2,
           ofield_m             ---> rationals_m,
           ofield_g             ---> rationals_g,
           ofield_c1            ---> rationals_c1,
           ofield_c2            ---> rationals_c2,
           ofield_id1           ---> rationals_id1,
           ofield_id2           ---> rationals_id2,
           ofield_inv1          ---> rationals_inv1,
           ofield_inv2          ---> rationals_inv2,
           ofield_ring          ---> rationals_ring,
           ofield_ringwid       ---> rationals_ringwid,
```

```
ofield_cringwid       ---> rationals_cringwid,
ofield_ag             ---> rationals_ag,
ofield_id             ---> rationals_id,
ofield_field          ---> rationals_field,
ofield_order          ---> rationals_order]).
```

The final argument copies the theories that have been interpreted into **ofield** into subtheories of **rational_ofield** under the renamings provided. For example, **ofield** has an interpreted **semigroup** subtheory **ofield_sg1**. A new subtheory, **rationals_sg1**, of **rational_ofield** is created by copying **ofield_sg1** and renaming its symbols as indicated by the first argument. As pointed out by an anonymous referee, this listing of subtheory renamings breaks down the modularity of the theories as the **rationals_ofield** theory must have knowledge of the subtheories of **ofield** theory.

This interpretation forms twenty-nine postulates in the **rationals_ofield** theory, all of which must be proved to ensure that the interpretation is valid. The proofs vary in complexity, yet even simple proofs, e.g., $1_{\mathbb{Q}} : \mathbb{Q}$ (where $1_{\mathbb{Q}}$ is the rational one), can require a significant number of proofs steps.

After the rationals have been created, the reals are modelled as equivalence classes of Cauchy sequences. Given sequences of rationals, `rationals_seq`, other standard rational operators and constants, sequence indexing '`at`', and universal quantification '`all`', Cauchy sequences are defined using standard Ergo functions indexed by the naturals:

```
define cauchy_seq ===
   set_of w:rationals_seq
     all e:rationals_set (e q> qzero =>
           ex ne:nats
               all m:nats all n:nats (m > ne and n > ne =>
                  qabs((w at m) q- (w at n)) q< e)).
```

After defining the equivalence classes, '`cequivclass`', as shown in Section 4.1, the reals are defined:

```
define reals ===
   set_of w:power(cauchy_seq)
      (ex an:cauchy_seq w = cequivclass(an)).
```

The ordered field properties are then interpreted into the theory of the reals in a manner identical to that used for the rationals above. The justifications that the interpretations are valid follow standard Cauchy sequence development proofs.

To complete the construction, the reals are shown to form a complete ordered field by proving that the supremum property holds.

Finally, since Ergo is a window inference based theorem prover, opening rules for the subterms of each new operator are required. Typically, the opening rules added allow subterms to be altered under an equivalence relation.

Hence the addition of an appropriate opening rule allows the equivalence class corresponding to a Cauchy sequence r_1 to be replaced by the equivalence class corresponding to another Cauchy sequence r_2 provided r_1 is related to r_2 by $\rho_{\mathbb{Q}^\omega}$.

4.3 Tool Design Considerations

There are many facets of Ergo that influenced the design of the reals theory. Some of the considerations that all tools face are:

Operator Ambiguities When dealing with multiple theory interpretations, where the same operator is used in more than one theory, the question arises as to which theory the operator belongs to. In Ergo, the ambiguities may be resolved by prefixing the operator with the theory name. Hence, to use ordered field 'addition', the addition operator is prefixed with the name of theory of ordered fields:

```
ofield.+
```

In comparison, to use real addition, the operator is prefixed with the name of the theory of reals:

```
reals.+
```

This solution is considered suboptimal. It would be better to be able to provide a default which is used when the operator is not prefixed. Ergo's macro facilities (abbreviations) could be used to provide the defaults. However, this would require reorganisation of the standard Ergo theories to avoid name clashes.

Specific versus generic syntax At some stage in the theory development, it is necessary to use theory identifier remapping to change from the generality indicated by the identifier '*op*' to the typical syntax of the reals, e.g., '+'. It could be argued that this remapping should be left as late as possible. That is, the use of specialised syntax may lead to an incorrect assumption that the semantics of real arithmetic is the only applicable domain. This should be weighed against the readability of the theories. That is, the increasing complexity of the theorems at this level makes it difficult for the maintainers to decode them into a well-known and understood domain so proof can be accomplished. The idea that the well-known domain of the reals provides enhanced readability of the theories was favoured.

Macro-interpretation One fault of Ergo's macro (abbreviation) facilities is that they are not remappable when the theory is interpreted. Consequently, when a theory with an abbreviation is instantiated, each abbreviation must be manually copied and altered using remappings. For example, the **ofield** theory contains an abbreviation for the binary subtraction operator.

$$T - R \mathrel{\widehat{=}} T + -R$$

When the **ofield** theory is instantiated into the **rational** theory, the abbreviation is not transferred. Thus the abbreviation must be duplicated in the **rational** theory.

$$T -_{\mathbb{Q}} R \mathrel{\widehat{=}} T +_{\mathbb{Q}} -_{\mathbb{Q}} R$$

Ease of use To aid ease of use of the theory, a mapping from the integers into the rationals is provided. Additionally, a mapping from the rationals into the reals is also provided. These mappings ease the construction of literal rational and real numbers. An alternative approach is to define a subset of the reals corresponding to the rationals, and a subset of those rationals corresponding to the integers. This would require showing the rationals subset either satisfies the axioms of the rationals, or that it is isomorphic to the model of the rationals given earlier. A similar proof would be required for the integers.

Ergo can also be customised with facilities known as oracles. Oracles make reasoning about expressions involving literal rationals and reals easier. For instance, an appropriate oracle can be used to transform the rational expression $[(1,2)]_{\rho_s} +_{\mathbb{Q}} [(1,2)]_{\rho_s}$ automatically into $[(1,1)]_{\rho_s}$.

Oracles could also be used to construct infinitely long literal rationals and (a subset of) real numbers using positional expansions automatically.

5 Comparison with Previous Work

Harrison has developed a 'scaling' technique with which he produced a mechanisation of the reals using Cauchy sequences [3]. The aim of the technique is to "fast-track" development by skipping the construction of the rationals. Rationals are created later as a subset of the reals. Effectively, Harrison follows the Cauchy sequences method but 'scales up' the Cauchy sequences to use naturals rather than rationals as their terms [3, p.16]. His development is equivalent to using Cauchy sequences, in this case with indices starting from one, that converge at a rate of $O(1/n)$ to a real number R, that is, a Cauchy sequence, w, that satisfies,

$$\exists\, B : \mathbb{N} \bullet \forall\, n : \mathbb{N}_1 \bullet |w[n] - R| < B/n$$

If we consider w alone, it satisfies,

$$\exists\, B : \mathbb{N} \bullet \forall\, n, m : \mathbb{N}_1 \bullet |w[n] - w[m]| < B/n + B/m$$

He then uses a sequence, a, of natural numbers in place of the rational sequence w, with the relationship $w[i] = a[i]/i$. The sequence a satisfies,

$$\exists\, B : \mathbb{N} \bullet \forall\, n, m : \mathbb{N}_1 \bullet |a[n]/n - a[m]/m| < B/n + B/m$$

Multiplying both sides by nm, we obtain:

$$\exists\, B : \mathbb{N} \bullet \forall\, n, m : \mathbb{N}_1 \bullet |m * a[n] - n * a[m]| < B * (m + n)$$

Consequently, we have obtained a constraint for Cauchy sequences on natural numbered terms, bypassing the need for construction of the rationals.

In a practical context, the avoided work is a benefit which outweighs the drawbacks of non-compliance with classical techniques and a slight complication in construction. If we used Harrison's 'scaled-up' approach, theory interpretation would still be beneficial to reduce theory development and promote reuse.

Harrison has generalised equivalence classes and produced an innovative implementation that automatically lifts the operators and first order theorems of a theory to produce equivalence classes. For example, he uses the technique on the Cauchy sequence operators and theorems to form the positive real numbers. In contrast, the Ergo development explicitly defined the equivalence classes using Cauchy sequences and the equivalence relation. Subsequently, the real operators were defined and the field theorems interpreted. By using the 'lifting' technique, Harrison was able to avoid defining the real operators explicitly and proving (or interpreting) the field theorems. We have not investigated the possibility of using the 'lifting' technique in Ergo.

Both Harrison's HOL development and the Ergo theory use a definitional extension technique. This essentially means that a model of the reals is provided and the desired axioms are proven as properties of the model. The main benefit of this technique is the increased confidence in the development. The model is used as a consistency check. An alternative approach would be to simply provide a set of axioms which characterise the reals.

6 Summary

Two aims have been achieved through this project. Most importantly, the project has used theory interpretation for a mechanised definitional extension development of the reals. The algebraic structure hierarchy is quite modular and open to exploitation by theory reuse mechanisms.

Secondly, the project is a case study evaluating the practical use of Ergo's theory interpretation facilities. In this regard the project has been quite successful. The theory interpretation facilities have been shown to be beneficial in their ability to reduce theory development and promote reuse. As discussed on page 5, using these facilities means that fifteen theories replace approximately forty-eight theories. Additionally, the areas in which the theory interpretation facilities are not so mature have been identified.

While Ergo was chosen, for familiarity reasons, for our work, this paper is applicable for any theorem prover that provides interpretation/instantiation facilities. It should also prove useful to those contemplating the inclusion of,

or reviewing the implementation of mechanised theory reuse facilities.

Acknowledgements

The authors would like to acknowledge Ray Nickson for his initial contributions to the project, Colin Fidge who proof read the extended version of the paper, and the anonymous referees for their comments and corrections. Additionally, the project was financially sponsored by ARC Large Grant A49702415: *Efficient Development of Verified Concurrent Real-Time Programs through Tool Support.*

References

[1] Billington, E., D. Donovan, B. Jones, S. Oates-Williams and A. Street, "Discrete Mathematics Logics and Structures, 2nd edition," Longman Cheshire Pty Ltd, 1993, second edition.

[2] Hamilton, N., R. Nickson, O. Traynor and M. Utting, *Interpretation and instantiation of theories for reasoning about formal specifications*, in: *Australian Computer Science Conference (ACSC'97)*, 1997, pp. 37–45.
URL `http://svrc.it.uq.edu.au/Bibliography/svrc-tr.html?96-2\%1`

[3] Harrison, J., "Theorem Proving with the Real Numbers," Ph.D. thesis, Churchill College, University of Cambridge (1996).

[4] Hayes, I. and M. Utting, *Coercing real-time refinement: A transmitter*, in: *BCS-FACS Northern Formal Methods Workshop, Electronic Workshops in Computing* (1997).

[5] Nickson, R., O. Traynor and M. Utting, *Cogito Ergo Sum: Providing structured theorem prover support for specification formalisms*, in: *Proceedings of the Nineteenth Australasian Computer Science Conference (ACSC'96)*, 1996, pp. 149–158.

[6] Robinson, P. and J. Staples, *Formalizing a hierarchical structure of practical mathematical reasoning*, Journal of Logic and Computation **3** (1993), pp. 47–61.

[7] Robinson, P. J., *Qu-Prolog 4.2 user guide*, Technical Report 97-12, Software Verification Research Centre, School of Information Technology, The University of Queensland, Brisbane 4072, Australia (1997).
URL `http://svrc.it.uq.edu.au/Bibliography/svrc-tr.html?97-1\%2`

[8] Shield, J., I. Hayes and D. Carrington, *Mechanising the reals in an interactive theorem prover using theory interpretation*, Technical Report 98-20, Software Verification Research Centre, The University of Queensland (1998).
URL `/~www/db/submitted/tr98-20.ps.Z`

[9] Utting, M. and C. Fidge, *Refinement of infeasible real-time programs*, in: *Proceedings Formal Methods Pacific (FMP'97)* (1997), pp. 243–262.