Theory Exploration: a role for Model Theory?

Alan Smaill

University of Edinburgh

Abstract

There is an empirical claim that, when exploring a mathematical theory, after a succession of key results have been obtained, a point of equilibrium is reached where any query of interest can be resolved by routine reasoning from the results already established. Here is suggested some ways of thinking about the situation, in general. There are at least some situations where we can establish that all results (of a certain shape) will follow by routine reasoning from a small number of key properties. An example is described, and the significance for automated theory exploration discussed.

1 Introduction

In a series of papers, Bruno Buchberger has presented a view of algorithmically based theory exploration (see e.g. [3, 4, 5]). Important features are the distinction between "easy proving" and "hard proving", and a "spiral" progression involving introduction of new concepts, successive stages of working out the key consequences with "hard reasoning", until reaching a point of saturation; at this point the "hard reasoning" stops providing any new insights, and "easy reasoning" is sufficient for subsequent use of the theory. This last possibility can give rise to streamlined procedures (e.g. normalisers or decision procedures) based on the theorems established with hard reasoning.

Examples in [4] include:

Arithmetic Given the recursion equations for plus, use "hard" means (induction) to prove appropriate arithmetic laws; given the right laws, induction becomes unnecessary; an efficient procedure for deciding equality of arithmetic expressions involving plus and 0 can then be validated.

Simple Set Operations Given notions of set membership and definitions of $\cup, \cap$, use FOL reasoning to establish identities involving $\cup, \cap$. Given such boolean algebra properties, simple rewriting is sufficient for their use, e.g. via a boolean algebra simplifier.

Real Closed Fields Again start with axioms and FOL reasoning. After hard work to establish the delineability theorem [8]; subsequently inference can proceed without reasoning involving quantifiers, and a decision procedure is able to be validated.

From these examples, we see that the notions of "hard" and "easy" reasoning are relative; that they do not correspond to algorithmic complexity (the real-closed field decision procedure embodies "easy" reasoning, but is computationally very expensive); that the "easy" reasoning involves less search than "hard" reasoning.

Having established that a situation has been reached where "easy" reasoning is sufficient for a given purpose, there is still work to do if the aim is to give an algorithm that exploits this sort of reasoning. Such algorithms, whose accuracy depends on the previous reasoning steps, should be verified.

Putting this together, we have some examples with a family resemblance. What sorts of tools and techniques can be brought to bear in order to facilitate and if possible automate

aspects of this exploration process? In this paper is described an approach where model theory can be used to identify situations where it is known that "easy" reasoning is sufficient.

The next sections describe: (2) a presentation by Henkin of such a model-theoretic argument; (3) a discussion of the approach via normal forms for the same problem; (4) the details of the argument from Henkin; (5) a plan for proving other results of this form; (6) its realisation for another example; and (7) concluding discussion.

2 Henkin's presentation

We first look at the first of Buchberger's examples above, the case of addition over the natural numbers with 0.

This particular example (actually the version without the identity 0) is discussed in [9] :

> But now let us return to the equational theory of the number system $\mathcal{N}$. The fact that the commutative and associative laws form a base can also be proved by means of normal forms for the terms of $\mathcal{T} = (T, \oplus)$. However, the result about this base has a semantical significance which permits a completely different kind of proof. Namely, the indicated result is equivalent to the statement that every structure $\mathcal{A} = (A, o)$ which satisfies the commutative and associative laws — i.e., every commutative semi-group — will satisfy any equation which holds identically in $\mathcal{N}$. And this can be proved by showing that each commutative semi-group $\mathcal{A}$ is a homomorphic image of a subdirect power of $\mathcal{N}$.
>
> [9, p 92]

This means that if we have an equation formed just using + and variables, and if it is universally true in $\mathcal{N}$, the natural numbers, then we can prove it by equational reasoning, without induction, using associativity and commutativity as axioms. Taking equational reasoning as "easy" and reasoning with induction as "hard", this means that for goals of the appropriate kind, we know that "easy" reasoning is complete, that there is no need to use "hard" reasoning.

3 Normal forms in natural numbers

Before looking at the details of the model-theoretic argument, consider the approach via normal forms mentioned by Henkin.

Henkin mentions that the result for natural numbers can be got by defining a normal form for terms formed of variables and +. So, for example, we can rewrite by sorting the variables according to some order or other, and left associate both terms; that this preserves the denotation if the terms follows exactly from commutativity and associativity. It's easy enough to see that this defines unique normal forms.

What is also true is that if the two normal forms are not identical, then the equation is not universally true, so that this form of reasoning is complete. If we compare the two terms, and cancel pairs of variables from the left and right terms that are identical, we must come to a situation with a variable that appears in one term but not the other, so we can devise a simple countermodel.

Although the argument in this case is simple, it looks to be quite hard to generalise the second step in a way that might give automatic recognition, in other similar cases. Presumably

an inductive argument over the syntax of normal forms, using appropriate cancellation lemmas is available.

The use of model-theoretic ideas in relation to exploiting common properties in automated reasoning is described for example in [2, Sec 5]; as in Henkin's exposition, this involves consideration of the models of the underlying theories, and indeed the notions of commutativity, associativity and identity. The main focus is on unification in theories where equality is characterised by some combination of these properties. In contrast, the emphasis for us is to identify when a given theory. involving inductive definitions, is such that the identities of the theory are correctly axiomatised in this way.

An alternative, proof theoretic, approach to show that "easy" reasoning suffices, is to try to show that for every inductive proof, there is a corresponding equational one from the lemmas – this looks difficult in general.

For the first stage, the area of devising normal forms from an equationally presented theory has had a fair bit of attention, for example via completion procedures such as Knuth-Bendix (see [1, Ch 7]). Work on synthesis of decision procedures in terms of rewrite systems is also relevant [10].

On the other hand, there is little work on *automatically* showing completeness of axiomatisations with respect to a larger theory (the inductively defined theory of addition, here).

4 The model theoretic argument

Henkin also alludes to a different way of showing that associativity and commutativity form a base. Here is a reconstruction of that argument.

4.1 The proof

We want to show that every equation of the above form valid in $\mathcal{N}$ follows from associativity and commutativity. By completeness of first order logic, it is enough to show that any such equation is true in all models of associativity and commutativity (call this an AC-structure; this is traditionally called a commutative semigroup). There are some operations of building new models from old that preserve validity of atomic formulae (equations, here). If we can show that every AC-structure can be got from $\mathcal{N}$ by some such operations, that will be enough.

Three such operations are:

1. Taking products. If, for each $i \in I$, $\mathcal{A}_i = (A_i, +_i)$ is a set with a binary operation, then the product $\prod_{i\in I} \mathcal{A}_i$ is given as the set of sequences $\{ (x_i)_{i\in I} \mid x_i \in A_i \}$, with the binary operation defined pointwise, i.e.

$$(x_i)_{i\in I} + (y_i)_{i\in I} =_{def} (x_i +_i y_i)_{i\in I}.$$

   Any equation true in each of the $\mathcal{A}_i$ is going to be true in $\prod_{i\in I} \mathcal{A}_i$.

2. Taking a substructure. If we take a subset of the domain of a structure closed under the operations of the structure, then that will also preserve universally valid equations, since they simply have to be true for fewer interpretations of the variables.

3. Taking a homomorphic image (equivalently, taking a quotient). If there is a homomorphism $f : \mathcal{Y} \to \mathcal{Z}$ (i.e. if $\forall y_1, y_2 \in Y\ f(y_1 + y_2) = f(y_1) + f(y_2)$) and f is surjective, then any equation valid in $\mathcal{Y}$ is also valid in $\mathcal{Z}$.

Now the claim is that any AC-structure can be got from $\mathcal{N}$ via these operations. For an example, think of the real numbers modulo 1, i.e. the real interval $[0, 1[$ with "wrap-around" addition.

Suppose that $\mathcal{X} = (X, +)$ is an AC-structure.

First, take a copy of $\mathcal{N}$ for each element $x \in X$ to get $\prod_{x \in X} \mathcal{N}$. This is also an AC-structure.

Now take the substructure of the product whose domain is the set

$$\{ (n_x)_{x \in X} \mid n_x = 0 \text{ for all but finitely many } x \}.$$

This is closed under the + operation on the product, so is also an AC-structure; call it $\mathcal{S}$.

We can now find a homomorphism $f : \mathcal{S} \to \mathcal{X}$ as follows.

Set

$$f((n_x)_{x \in X}) \quad =_{def} \quad \sum_{x \in X} n_x.x$$

where the sum is defined since only a finite number of terms are non-zero. $n_x.x$ for n_x a natural number, and $x \in X$, means the n-fold addition of x to itself in X. Now it should be checked that f is a homomorphism.

Since each step preserves validity of equations, this shows that all equations valid in $\mathcal{N}$ are also valid in $\mathcal{X}$.

It is worth noting that the proof, extended to allow an identity, also shows that these consideration as also apply to equations over reals, rationals, complex numbers, using +, simply because these are commutative semi-groups.

4.2 Can we use this?

This looks like a harder proof, both to establish by hand, and to hope to automate, and there is little work on automation of such proofs. However there are quite a lot of such preservation theorems known (see [7]) that relate syntactic categories of axiomatisations to preservation under operations on structures. Building up a theory of how to construct appropriate structures would be a real challenge.

It will appear from the example in this paper that when induction is used in the proofs (when fully spelt out, over the syntax of formulae), its use is straightforward. At this stage, this is anecdotal evidence in favour of this approach.

5 Other examples, and a proof plan

Here are some similar situations, where we can expect the same approach to work:

1. Equations over naturals, using + and 0 (as above, with identity property).
2. Equations over naturals, using $\times$ and 1.
3. Equations of lists, with terms formed from *append*, *nil*, using associativity of append, and identity property (on both sides of append).
4. Equations of sets, using union and intersection (from boolean algebra characterisation, including distributivity).
5. ...

Let us look for an approach to the automation of proofs for these examples in the spirit of *proof plans* [6]; the idea is that search for proof is guided by an expectation of the overall shape of the proof, and a planning-based approach to the assembly of tactics that build a final verification. The shape of the strategy will be given informally here.

5.1 The task

Let us use the second example above; in fact a slight variant will simplify the presentation here. Use $\mathcal{N}_{>0}$ for the structure with domain the positive natural numbers, and the usual multiplication and multiplicative unit; the set of positive natural numbers is written as $N_{>0}$.

We will look at the claim applied to the positive natural numbers; the full claim as above follows easily, by noting separately that for such identities

$$\mathcal{N}_{>0} \models t(\vec{x}) = s(\vec{x}) \quad \text{iff} \quad \mathcal{N} \models t(\vec{x}) = s(\vec{x}).$$

This is done by simply considering the cases where a variable is assigned the zero value when the variable appears on both sides of the identity, and only on one side.

The claim we want to verify is the following:

> If $t(\vec{x}), s(\vec{x})$ are terms built from variables[1] and the constant 1 using $\times$, with the usual interpretation in the positive natural numbers $\mathcal{N}_{>0}$, and ass, com, id are the usual statements of associativity, commutativity and identity for this language, then
>
> $$\mathcal{N}_{>0} \models t(\vec{x}) = s(\vec{x}) \quad \text{iff} \quad ass, com, id \vdash t(\vec{x}) = s(\vec{x}).$$

The proof idea is to take any model $\mathcal{S} = (S, \times_S, 1_S)$ of ass, com, id, and show that it can be regarded as related with $\mathcal{N}_{>0}$ by a series of intermediate structures, each of which preserves the truth of such identities; this shows that any true identity is a semantic consequence of ass, com, id, and we can appeal to completeness of equational reasoning.

5.2 The Strategy

The task involves supplying particular roles so that the parts of the proof fit together as follows. Suppose given a structure $\mathcal{S}$ with the appropriate properties:

1. Take a product of $\mathcal{N}_{>0}$, indexed by the elements of S. Operations are defined pointwise; this preserves truth of identities.
2. Take a substructure of the product, by picking appropriates sequences $(n_s)_{s \in S}$ closed under multiplication. This will preserve all purely universally quantified statements.
3. Define a "transfer function" of $N_{>0}$ to the algebra $\mathcal{S}$, i.e. a function $. : N \times S \to S$. In the case of addition, this was simply defined $n \,.\, s$ as the n-fold sum of s with itself, giving the key property that $(n + m) \,.\, s = (n \,.\, s) +_S (m \,.\, s)$. On the assumption that the properties of the algebra should be carried over in the same style, we want an operation . such that:
$$\begin{aligned} (n \times m) \,.\, s &= (n \,.\, s) \times_S (m \,.\, s) \\ 1 \,.\, s &= 1_s. \end{aligned}$$
4. Finally, we get $\mathcal{S}$ as a homomorphic image of the substructure in part 2 by defining a homomorphism that uses the local operation defined in part 3.

This splits the problem up into selecting the right ingredients to fit this outline.

[1] not necessarily the same set of variables.

6 From Plan to Proof

Now we look at how the planned shape of proof can be fleshed out into a proof. The proof planning approach suggests using middle-out reasoning here – the missing ingredients do not have to be immediately provided, but place-holders in the form of meta-variables are used to allow planning to continue; the meta-variables are then incrementally instantiated as verification conditions are satisfied. For the present example, this step is currently done on paper — see [12, 11] for examples of middle-out reasoning in proof planning.

Forming the product structure – no choice here, this follows the first step from section 4.1.

Finding the substructure The earlier proof, replacing + with times, suggests looking at the families which are "nearly the identity", i.e. $(n_s)_{s\in S}$ where $n_s = 1$ for all but finitely many values. The proof obligation here is to check that such sequences are closed under multiplication, and that the identity element $(1)_{s\in S}$ is included. Let's call this structure $\mathcal{T}$.

The first proof obligation can be discharged by induction on the number of non-identity elements.

Defining an operation Apart from the unhelpful operation that maps everything to 1_S, it is not so easy to invent an operation here. Using primitive recursion directly gets nowhere; since it's the multiplicative structure that is our interest, we can take recursion via factorisation and build in the property we want. This form of definition is suggested by a form of recursion analysis, as in [12].

$$\begin{aligned} 1 \,.\, s &= 1_S \\ p \,.\, s &= ? \quad \text{(where } p \text{ is prime)} \\ (x \times y) \,.\, s &= (x \,.\, s) \times_S (y \,.\, s) \end{aligned}$$

where we need to supply the base case values, and check that our function is well-defined, in that it does not depend on the factorisation chosen.

The solution falls out:

$$\begin{aligned} 1 \,.\, s &= 1_S \\ p \,.\, s &= s \quad \text{(where } p \text{ is prime)} \\ (x \times y) \,.\, s &= (x \,.\, s) \times_S (y \,.\, s) \end{aligned}$$

so that $n \,.\, s$ is s to the power m, where m is the number of occurrences of prime divisors of n, including repeated factors. The proof obligation here requires us to show for example

$$x \times y = x' \times y' \;\rightarrow\; (x \,.\, s) \times_S (y \,.\, s) = (x' \,.\, s) \times_S (y' \,.\, s),$$

and itself uses prime/composite induction.

Defining the homomorphism Given the operation and the structure $\mathcal{T}$, we want $h : T \to S$ such that

1. $h((1)_{s\in S}) = 1_S$
2. $h((n_s)_{s\in S} \times (m_s)_{s\in S}) = h((n_s)_{s\in S}) \times_S h((m_s)_{s\in S})$, and

3. h is surjective.

A uniform way to use an operation in this situation is to take

$$h((n_s)_{s\in S}) = \prod_{s\in S} n_s \,.\, s.$$

We can then check the three conditions. Commutativity and associativity are needed to prove the second homomorphism property, via induction on the number of values in $(n_s)_{s\in S}$ that are not 1.

Thus the result is established.

7 Discussion

This looks reasonably hopeful for automation; we have a proof outline which can be expressed in a higher-order language, verified in general, and used to guide the choice of components in a given application. The proof planning framework provides control for instantiation of the needed components in a middle-out way. More work is needed to see how generally applicable this is, and to what extent in can be automated.

These results are interesting for search, because they tell us that once we have established a small number of inductive theorems in the object theory, there is no need for more inductive proofs when considering goals of a certain shape.

In contrast to other approaches, no assumption about decidability is made here; the restriction to easy reasoning improves the situation for search, even if the problem remains undecidable. The assumption here is that hard reasoning subsumes easy reasoning, i.e. that easy reasoning steps are a subset of the steps permitted in hard reasoning: in this case, we are left with a smaller search space. However, it does not give us any guarantee of finding shorter proofs – it may well be that there are short hard proofs, that require longer easy proofs.

On the other hand, this sort of result allows us to transfer decision procedures if we have them (e.g. of provability from a few algebraic properties) to other domains. These are interesting properties for modularisation of search.

References

[1] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[2] F. Baader and W. Snyder. Unification theory. In J. A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning, Volume 1*, volume I, chapter 8, pages 447–553. Elsevier, 2001.

[3] Bruno Buchberger. Algorithm-supported mathematical theory exploration: A personal view and strategy. In Bruno Buchberger and John A. Campbell, editors, *Artificial Intelligence and Symbolic Computation, 7th International Conference, AISC 2004*, number 3249 in Lecture Notes in Computer Science, pages 236–250. Springer, 2004.

[4] Bruno Buchberger. Mathematical theory exploration, 2006. Invited talk at IJCAR-06.

[5] Bruno Buchberger. Theory exploration versus theorem proving: Why automated theorem proving has little impact on mathematics. 2008.

[6] A. Bundy. A science of reasoning. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 178–198. MIT Press, 1991.

[7] C. C. Chang and H. J. Keisler. *Model Theory*. North-Holland, Amsterdam, 1973.

[8] George Collins. Quantifier elimination for real closed fields by cylindrical algebraic decompostion. In H. Brakhage, editor, *Automata Theory and Formal Languages: 2nd GI Conference Kaiserslautern, May 20–23, 1975*, volume 33 of *Lecture Notes in Computer Science*, pages 134–183, Berlin, 1975. Springer.

[9] L. Henkin. Algebraic aspects of logic: past, present, future. In *Colloque International de Logique*, pages 89–106, Paris, 1977. Éditions du Centre National de la Recherche Scientifique.

[10] Predrag Janicic and Alan Bundy. Automatic synthesis of decision procedures. In Manuel Kauers, Manfred Kerber, Robert Miner, and Wolfgang Windsteiger, editors, *Towards Mechanized Mathematical Assistants: 14th Symposium, Calculemus 2007*, volume 4573 of *Lecture Notes in Artificial Intelligence*, pages 80–93. Springer, 2007.

[11] M. Johansson, L. Dixon, and A. Bundy. Dynamic rippling, middle-out reasoning, and lemma discovery. In *Walther Festschrift*, number 6463 in Springer, pages 102–116. LNCS, 2010.

[12] I. Kraan, D. Basin, and A. Bundy. Middle-out reasoning for synthesis and induction. *Journal of Automated Reasoning*, 16(1–2):113–145, 1996. Also available from Edinburgh as DAI Research Paper 729.