Studying Algebraic Structures using Prover9 and Mace4

Rob Arthan* & Paulo Oliva†

August 14, 2019

Abstract

In this chapter we present a case study, drawn from our research work, on the application of a fully automated theorem prover together with an automatic counter-example generator in the investigation of a class of algebraic structures.

We will see that these tools, when combined with human insight and traditional algebraic methods, help us to explore the problem space quickly and effectively. The counter-example generator rapidly rules out many false conjectures, while the theorem prover is often much more efficient than a human being at verifying algebraic identities.

The specific tools in our case study are Prover9 and Mace4; the algebraic structures are generalisations of Heyting algebras known as hoops. We will see how this approach helped us to discover new theorems and to find new or improved proofs of known results. We also make some suggestions for how one might deploy these tools to supplement a more conventional approach to teaching algebra.

1 Introduction

Prover9 is an automated theorem prover for first order logic [8]. The companion program Mace4 searches for finite models of first order theories. Tools such as these have been used to attack problems in algebra for many years.A headline result was McCune's use of a predecessor of Prover9 to find the first proof of the Robbins conjecture [9]. It is noteworthy in connection with the present chapter that the first proof relied on reductions of the conjecture that were originally proved by Winker [11] using a mixture of human reasoning and automated proof search. Several people worked on human readable accounts of the machine-generated proof. [5, 4].

It is the goal of this chapter to illustrate an approach that we have found very useful in research on algebraic structures: we develop human readable proofs of new results by mixing automated proof search and human analysis of the proofs found by Prover9 and of examples found by Mace4. We propose that guided use of such tools on known examples could also be of benefit in teaching algebra.

*School of Electronic Engineering and Computer Science, Queen Mary University of London, Mile End Road, London E1 4NS, UK. rda@lemma-one.com

†School of Electronic Engineering and Computer Science, Queen Mary University of London, Mile End Road, London E1 4NS, UK. p.oliva@qmul.ac.uk (corresponding author)

In Section 1.1 we illustrate the use of the tools taking the theory of *semilattices* as a simple example. In Section 1.2 we give an introduction to the class of algebraic structures called *hoops* and apply Prover9 and Mace4 to an investigation of these algebraic structures at the level of a possible undergraduate project.

In Section 2 of this chapter, we present some results from our own research obtained with the assistance of Prover9 and Mace4. Surprisingly, Prover9's machine-generated proofs can be tractable and useful artefacts. Human insight allows us to extract new abstractions and further conjectures, leading to an iterative interactive process for developing the theory and producing a human-oriented account of the proofs.

The examples in this chapter are supported by a set of Mace4 and Prover9 scripts, available from `http://www.eecs.qmul.ac.uk/~pbo/PTMRT/`. There the reader can also find information about obtaining Prover9 and Mace4 and instructions for running the scripts. The examples are organised into two separate folders, one for Section 1 and one for Section 2. In these sections, references to named files refer either to the scripts or the output files produced by running them in the corresponding folder.

As a final introductory remark, we would like to stress that we are concerned here with applications of tools. Applications invariably suggest desirable new features or new tools, but we are concerned here with the development of mathematical knowledge, either in a research or a teaching context, by exploiting potential synergy between human capabilities and the capabilities of tools that are available now. We believe that this kind of synergy will be required at every stage of technological advancement and that to learn this from practical experience is a worthwhile achievement in its own right. We confine our thoughts on possible future developments to the tools to our concluding remarks.

1.1 Using Prover9 and Mace4

In this section we will introduce Prover9 and Mace4 using the theory of semilattices as an example.

Recall that a semilattice can be defined as a set equipped with a least upper bound operation on pairs of elements: i.e., a structure $(S, \cup)$ where the binary operation $\cup$ is associative, commutative and idempotent, i.e., it satisfies:

$$x \cup (y \cup z) = (x \cup y) \cup z$$
$$x \cup y = y \cup x$$
$$x \cup x = x.$$

Prover9 and Mace4 use a common syntax for first-order logic. In this syntax, we can formalise the semilattice axioms as follows (see file `semilattice.ax`):

```
op(500, infix, "cup").
formulas(assumptions).
   (x cup y) cup z = x cup (y cup z).
   x cup y = y cup x.
   x cup x = x.
end_of_list.
```

Here the first line declares that we will use `cup` as an infix operator. This is followed by the axioms of our theory (referred to as "assumptions" by Prover9 and its documentation) using `cup` in place of ∪[1]. By convention, the letters `x`, `y` and `z` denote variables that are implicitly universally quantified.

We can now ask Prover9 to prove theorems based on these assumptions. As a very simple example we can formulate the following goal[2]:

```
formulas(goals).
   x cup (x cup x) = x.
end_of_list.
```

If we now execute the following command:

```
   Prover9 -f  semilattice.ax sl-pr1.gl
```

then Prover9 will almost instantaneously respond with the pleasing message "`THEOREM PROVED`". Its output also includes a detailed linear representation of the proof[3]:

```
1 x cup (x cup x) = x # label(non_clause) # label(goal).  [goal].
2 x cup x = x.  [assumption].
3 c1 cup (c1 cup c1) != c1.  [deny(1)].
4 $F.  [copy(5),rewrite([4(4),4(3)]),xx(a)].
```

At first glance this looks daunting, but with a little work it is possible to gain insight from it. The Prover9 proofs are always presented as proofs by contradiction: after stating the goal and the assumptions that will be used to prove it, Prover9 denies the goal by asserting the existing of a constant `c1` that does not satisfy the equation we are trying to prove (`!=` is the Prover9 syntax for ≠). In general, the line that denies the goal will be followed by a sequence of steps each comprising a formula that can be deduced by applying logical inference rules to formulas given in earlier steps. In this case, there is just one step in this sequence, in which some rewriting has arrived at the desired contradiction `$F` (Prover9 syntax for falsehood). In each inference step, the formula is followed by a justification giving a precise description of the inference rules and how they have been applied. There is a program `prooftrans` supplied with Prover9 that can be used to perform some useful transformations on proofs (we have already used it in generating the proof as shown above to renumber the steps in consecutive order). In this case, we can ask for the rewriting steps to be expanded giving:

```
4A c1 cup c1 != c1.  [para(2(a,1),3(a,1,2))].
4B c1 != c1.  [para(2(a,1),4A(a,1))].
4 $F.  [copy(4B),xx(a)].
```

It then quickly finds that repeated rewriting with the idempotency law (the assumption, i.e., axiom stated in step 2) gives the false (`$F`) conclusion that `c1` is not equal to itself.

[1] Prover9 input uses only ASCII characters.
[2] See goal script `sl-pr1.gl`.
[3] See output file `sl-pr1.txt`.

The justifications in the Prover9 proofs look complicated because they encode a very detailed description of how each inference rule has been applied. In trying to understand the proof, one can generally ignore most of this detail. The rule names `copy`, `para`[4]etc. are followed by a list of parameters in brackets identfying the steps that provided the inputs (antecedents) to the rule the list of letters and numbers in brackets that come immediately after the step number identify exactly how the rule was applied to the formula of that step, but in most cases that is obvious and this information can be ignored.

In a semilattice, one defines a relation $\geq$ by

$$x \geq y \quad \Leftrightarrow \quad x \cup y = x.$$

We can formalise this definition in Prover9 syntax as follows (see file `sl-ge-def.ax`):

```
formulas(assumptions).
   x >= y <-> x cup y = x.
end_of_list.
```

We can now get Prover9 to prove some more interesting properties. E.g. the transitivity of $\geq$[5]:

```
formulas(goals).
   x >= y & y >= z -> x >= z.
end_of_list.
```

If we execute the command:

```
   Prover9 -f  semilattice.ax sl-ge-def.ax sl-trans.gl
```

then Prover9 will again quickly respond with "`THEOREM PROVED`". In this case the proof it finds has 20 steps[6]. As always it is a proof by contradiction, but with a little analysis it is easy to extract the elements of a human-readable direct proof from it. As we will see later in this chapter, it is not always easy to extract human-readable proofs from the Prover9 output, but analysis of the machine-generated proofs often leads to useful insights.

The reader may well ask what happens if we ask Prover9 to prove a false conjecture. What if we were to conjecture that the ordering relation on a semilattice is a total order? We would express this goal in Prover9 as follows (see file `sl-total.gl`):

```
formulas(goals).
   x >= y | y >= x.
end_of_list.
```

[4] `para` stands for "paramodulation", an inference rule that performs a form of equational reasoning generalising the usual notion of using an equation to rewrite a term within formula.

[5] See goal script `sl-trans.gl`.

[6] See output file `sl-trans.txt`.

Here "|" denotes disjunction, and, recalling that variables are implicitly universally quantified, in textbook logical notation the goal is $\forall x \forall y (x \geq y \vee y \geq x)$. If we run Prover9 on this goal it will necessarily fail to deduce a contradiction: it will either fail to terminate or terminate without finding a proof (either because no new formulas can be generated or because it has reached a user-specified bound on execution time or memory usage). In this case the companion program Mace4 does something much more useful. If we execute the command:

```
mace4 -p 1 -f semilattice.ax sl-ge-def.ax sl-total.gl
```

then the Mace4 program will search for a finite semilattice that provides a counter-example to our false conjecture. It quickly finds one and in its log of the search process it prints out the counter-example as follows:

```
cup :
     | 0 1 2
   --+------
   0 | 0 2 2
   1 | 2 1 2
   2 | 2 2 2

>= :
     | 0 1 2
   --+------
   0 | 1 0 0
   1 | 0 1 0
   2 | 1 1 1
```

Here we see that Mace4 uses $0, 1, 2 \ldots$ to represent the elements of the model, which in this case has size 3. The model is the smallest example of a semilattice that is not totally ordered. It is made up of two incomparable atoms 0 and 1 together with their least upper bound $2 = 0 \cup 1$.

In conjunction with two companion programs `isofilter` and `interpformat`, Mace4 can enumerate all the models of a given size. The `Makefile` contains the commands to do this for all semilattices with at most 5 elements. This is particularly useful when investigating more complex algebraic structures as generating examples by hand is often error prone, particularly if associative operators are involved as associativity is time-consuming to check.

1.2 Investigating the algebraic structure of hoops

In Section 1.1, we took semilattices as our running example for reasons of simplicity. However, semilattices are a little too simple to demonstrate the real power of tools such as Prover9 and Mace4. In this section we introduce the algebraic structures called hoops [1, 2, 3] that will provide the running example for the rest of the paper. Hoops are a generalisation of Heyting algebras (used in the study of intuitionistic logic[7]). They are

[7] In a Heyting algebra one normally uses $x \to y$ for $y \ominus x$, and $x \wedge y$ for $x \oplus y$.

of considerable interest, e.g., in connection with Łukasiewicz logic and fuzzy logic, and there are many difficult open problems concerning them.

These structures were originally investigated, first by Bosbach [2], and independently by Büchi and Owens [3]. In this section, we present an elementary investigation of hoops using Prover9 and Mace4, and indicate how one might put these tools to use at the level of an undergraduate project assignment.

A hoop[8] is a structure $(H, 0, 1, \oplus, \ominus)$ satisfying the following axioms:

$$x \oplus (y \oplus z) = (x \oplus y) \oplus z \tag{1}$$
$$x \oplus y = y \oplus x \tag{2}$$
$$x \oplus 0 = x \tag{3}$$
$$x \ominus x = 0 \tag{4}$$
$$(x \ominus y) \ominus z = x \ominus (y \oplus z) \tag{5}$$
$$x \oplus (y \ominus x) = y \oplus (x \ominus y) \tag{6}$$
$$0 \ominus x = 0 \tag{7}$$
$$x \ominus 1 = 0 \tag{8}$$

The example scripts in the supporting material for this section contain a Prover9 formalisation of the hoop axioms (file `hoop-eq-ax`) and various goals for Prover9 and Mace4.

Axioms (1), (2) and (3) are very familiar as the axioms for a commutative monoid with binary operation $\oplus$ and identity element 0. Axioms (4) and (5) are reminiscent of properties of subtraction in a commutative group, but the remaining axioms are less familiar.

Axiom (6) says that the operation $\cup$ defined by $x \cup y = x \oplus (y \ominus x)$ is commutative. Using axioms (3) and (4) we see that this operation is also idempotent. We might conjecture that $\cup$ is also associative, so that it makes any hoop into a semilattice, and Prover9 will readily prove this for us[9]. The proof is short but intricate. This semilattice structure induces an ordering on a hoop which turns out to be equivalent to defining $x \geq y$ to hold when[10] $y \ominus x = 0$.

Using Mace4, we can quickly generate examples of hoops. Tables 1 and 2 are based on the Mace4 output and show all hoops with between 2 and 4 elements. Inspection of these tables is very instructive, particularly if one looks at the interactions between the order structure and the algebraic operations.

In all cases, one notes that the elements in the rows and columns of the operation tables for $\oplus$ are in increasing order; in the tables for $\ominus$ the elements in the rows are in decreasing order while the elements in the columns are in increasing order. This suggests the conjecture that $\oplus$ is monotonic in both its operands, while $\ominus$ is monotonic in its right operand and anti-monotonic in its left operand. Prover9 quickly proves this for us[11].

[8] Strictly speaking this is a *bounded* hoop: an unbounded hoop omits the constant 1 and axiom (8). We are only concerned with bounded hoops in this book, so for brevity, we drop the word "bounded".

[9] See output file `hp-semilattice.txt`.

[10] See output file `hp-ge-sl.txt`.

[11] See output file `hp-plus-mono.txt`, `hp-sub-mono-left.txt` and `hp-sub-mono-right.txt`.

Ordering	Operation Tables		Name
$0 < 1$	$\begin{array}{c\|cc} \oplus & 0 & 1 \\ \hline 0 & 0 & 1 \\ 1 & 1 & 1 \end{array}$	$\begin{array}{c\|cc} \ominus & 0 & 1 \\ \hline 0 & 0 & 0 \\ 1 & 1 & 0 \end{array}$	$\mathbf{L}_2$
$0 < a < 1$	$\begin{array}{c\|ccc} \oplus & 0 & a & 1 \\ \hline 0 & 0 & a & 1 \\ a & a & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}$	$\begin{array}{c\|ccc} \ominus & 0 & a & 1 \\ \hline 0 & 0 & 0 & 0 \\ a & a & 0 & 0 \\ 1 & 1 & a & 0 \end{array}$	$\mathbf{L}_3$
$0 < a < 1$	$\begin{array}{c\|ccc} \oplus & 0 & a & 1 \\ \hline 0 & 0 & a & 1 \\ a & a & a & 1 \\ 1 & 1 & 1 & 1 \end{array}$	$\begin{array}{c\|ccc} \ominus & 0 & a & 1 \\ \hline 0 & 0 & 0 & 0 \\ a & a & 0 & 0 \\ 1 & 1 & 1 & 0 \end{array}$	$\mathbf{L}_2 \frown \mathbf{L}_2$

Table 1: Hoops of order 2 and 3

Axiom (5) suggests (and inspection of the tables supports) the conjecture that for any x, y and z, $z \geq x \ominus y$ iff $z \oplus y \geq x$. This property, an analogue of one of the laws for manipulating inequalities in an ordered commutative group, is known as the *residuation* property and is quickly proved by Prover[12].

A structure for the signature $(0, 1, \oplus, \ominus, \geq)$ such that $(0, \oplus, \geq)$ is an ordered commutative monoid with least element 0, greatest element 1 and satisfying the residuation axiom:

$$z \geq x \ominus y \Leftrightarrow z \oplus y \geq x$$

is known as a (bounded) *pocrim*. One might conjecture that any pocrim is a hoop. However this conjecture is false, if we ask Mace4 to enumerate small pocrims[13], it finds 2 pocrims with 4 elements that are not hoops, as shown in Table 3. The residuation property is strictly weaker than axiom (6). Inspection of the operation tables reveals the weakness: in a hoop, if $x \geq y$ then $x = y \oplus (x \ominus y)$, but in a pocrim, even when $x \geq y$, we can have $x < y \oplus (x \ominus y)$: in the first example in Table 3, $x \oplus y = 1$ unless one of x and y is 0. However, the axiomatisation of hoops via the pocrim axioms together with axiom (6) is often more convenient and intuitive than the purely equational axiomatisation.

Inspection of Tables 1 and 2, shows that for each n, there is a hoop of order n that is linearly ordered and generated by its least nonzero element, in the sense that if a is the least nonzero element, the other nonzero elements are $a \oplus a$, $a \oplus a \oplus a$, etc. For any n, let us define[14]

$$\mathbf{L}_n = (\{0, \frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\}, 0, 1, \oplus, \ominus)$$

[12] See output file `hp-res.txt`.

[13] See output file `pc-egs.txt`.

[14] We call these hoops $\mathbf{L}_n$ in honour of Łukasiewicz [7] whose multi-valued logics have a natural semantics with values in these hoops.

Ordering	Operation Tables		Name
$0 < a < b < 1$	$\begin{array}{c\|cccc} \oplus & 0 & a & b & 1 \\ \hline 0 & 0 & a & b & 1 \\ a & a & a & b & 1 \\ b & b & b & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$ $\begin{array}{c\|cccc} \ominus & 0 & a & b & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ a & a & 0 & 0 & 0 \\ b & b & b & 0 & 0 \\ 1 & 1 & 1 & b & 0 \end{array}$	$\mathbf{L}_2 \frown \mathbf{L}_3$
$0 < a < b < 1$	$\begin{array}{c\|cccc} \oplus & 0 & a & b & 1 \\ \hline 0 & 0 & a & b & 1 \\ a & a & b & 1 & 1 \\ b & b & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$ $\begin{array}{c\|cccc} \ominus & 0 & a & b & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ a & a & 0 & 0 & 0 \\ b & b & a & 0 & 0 \\ 1 & 1 & b & a & 0 \end{array}$	$\mathbf{L}_4$
$0 < a < b < 1$	$\begin{array}{c\|cccc} \oplus & 0 & a & b & 1 \\ \hline 0 & 0 & a & b & 1 \\ a & a & b & b & 1 \\ b & b & b & b & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$ $\begin{array}{c\|cccc} \ominus & 0 & a & b & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ a & a & 0 & 0 & 0 \\ b & b & a & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{array}$	$\mathbf{L}_3 \frown \mathbf{L}_2$
$0 < a < b < 1$	$\begin{array}{c\|cccc} \oplus & 0 & a & b & 1 \\ \hline 0 & 0 & a & b & 1 \\ a & a & a & b & 1 \\ b & b & b & b & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$ $\begin{array}{c\|cccc} \ominus & 0 & a & b & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ a & a & 0 & 0 & 0 \\ b & b & b & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{array}$	$\mathbf{L}_2 \frown \mathbf{L}_2 \frown \mathbf{L}_2$
$0 < a, b < 1$ $a \not\leq b$ $b \not\geq a$	$\begin{array}{c\|cccc} \oplus & 0 & a & b & 1 \\ \hline 0 & 0 & a & b & 1 \\ a & a & a & 1 & 1 \\ b & b & 1 & b & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$ $\begin{array}{c\|cccc} \ominus & 0 & a & b & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ a & a & 0 & a & 0 \\ b & b & b & 0 & 0 \\ 1 & 1 & b & a & 0 \end{array}$	$\mathbf{L}_2 \times \mathbf{L}_2$

Table 2: Hoops of order 4

Ordering	Operation Tables

$0 < a < b < 1$

$\oplus$	0	a	b	1
0	0	1	1	1
a	a	1	1	1
b	b	1	1	1
1	1	1	1	1

$\ominus$	0	a	b	1
0	0	0	0	0
a	a	0	0	0
b	b	a	0	0
1	1	a	a	0

$0 < a < b < 1$

$\oplus$	0	a	b	1
0	0	a	b	1
a	a	a	1	1
b	b	1	1	1
1	1	1	1	1

$\ominus$	0	a	b	1
0	0	0	0	0
a	a	0	0	0
b	b	b	0	0
1	1	b	a	0

Table 3: Pocrims that are not hoops

where $x \oplus y = \mathsf{min}(x + y, 1)$ and $x \ominus y = \mathsf{max}(x - y, 0)$. Then $\mathbf{L}_n$ is a linearly ordered hoop generated by its least non-zero element $\frac{1}{n-1}$. Copies of $\mathbf{L}_n$ for various n often appear in other hoops. Inspection of the first hoop in Table 2 shows that, the subset $\{0, b, 1\}$ comprises a subhoop isomorphic to $\mathbf{L}_3$, while the subset $\{0, a\}$ is isomorphic to $\mathbf{L}_2$ viewed as a structure for the signature $(0, \oplus, \ominus)$ (i.e., ignoring the fact that $a \neq 1$). This suggests a way of constructing new hoops from old: given hoops $\mathbf{H}_1$ and $\mathbf{H}_2$ with underlying sets H_1 and H_2, we can form what is called the *ordinal sum*, $\mathbf{H}_1 \frown \mathbf{H}_2$ of the two hoops whose underlying set is the disjoint union $H_1 \sqcup (H_2 \setminus \{0\})$, with 0 (resp. 1) given by $0 \in H_1$ (resp. $1 \in H_2$) and with the operation tables defined to extend the operations of $\mathbf{H}_1$ and $\mathbf{H}_2$ so that for $h_1 \in H_1$ and $h_2 \in H_2$, $h_1 \oplus h_2 = h_2$, $h_1 \ominus h_2 = 0$ and $h_2 \ominus h_1 = h_2$. The column headed "Name" in Tables 1 and 2 gives an expression for each hoop as an ordinal sum or product of the hoops $\mathbf{L}_n$.

Linearly ordered hoops are of particular importance. We can see at a glance from Tables 1 and 2 that there is 1 linearly ordered hoop of order 2 and 2 of order 3 (and these are the only hoops of these orders), while there are 4 linearly ordered hoops of order 4 (and just one other). Mace4 tells us[15] that there are 8 linearly ordered hoops of order 5 (and two others[16]). This leads us to the following theorem. The proof of this theorem proceeds by induction and Prover9 cannot be expected to find a proof automatically, but if, informed by the examples provided by Mace4, we set up the framework for the inductive proof, then Prover9 will help us with the low-level details.

Theorem 1.1 *For each $n \geq 2$, there are 2^{n-2} isomorphism classes of linearly ordered hoops of order n.*

Proof: We claim that any linearly ordered hoop of order n is isomorphic to an ordinal sum $\mathbf{L}_{m_1} \frown \ldots \frown \mathbf{L}_{m_k}$ for some k, $m_1, \ldots, m_k \geq 2$ such that $m_1 + \ldots + m_k - k +$

[15] See output file `hp-linear-egs.txt`.
[16] See output file `hp-egs.txt`.

$1 = n$. It is easy to see that two such ordinal sums $\mathbf{L}_{m_1} \frown \ldots \frown \mathbf{L}_{m_k}$ and $\mathbf{L}_{n_1} \frown \ldots \frown \mathbf{L}_{n_l}$ are isomorphic iff $k = l$ and $m_i = n_i$, $1 \leq i \leq k$. Moreover the sequences $\langle m_1, \ldots, m_k \rangle$ indexing these ordinal sums are in one-to-one correspondence with the subsets of $\{1, \ldots, n-2\}$ via:

$$\langle m_1, \ldots, m_k \rangle \mapsto \{m_1 - 1, m_1 + m_2 - 2, \ldots, m_1 + \ldots + m_{k-1} - k + 1\},$$

Hence there are, indeed, 2^{n-2} such hoops. We have only to prove the claim which follows immediately by induction from the observation that any finite linearly ordered hoop $\mathbf{H}$ is isomorphic to $\mathbf{L}_m \frown \mathbf{K}$ for some m and some subhoop $\mathbf{K}$ of $\mathbf{H}$. This observation may be proved by considering the subhoop generated by the least non-zero element of $\mathbf{H}$ using the fact that (in any hoop), if $x + x = x$ and $y \geq x$, then $x + y = y$, which Prover9 will readily verify for us[17].

This theorem and its proof are a nice example of synergy between automated methods and the traditional approach: Mace4 provides examples that suggest a general conjecture and indicate a possible line of proof by induction. While it is not able to automate the inductive proof[18], Prover9 can help us fill in tricky algebraic details.

We encourage readers interested in using tools such as Prover9 and Mace4 in undergraduate teaching to use the example scripts we have provided to inform the design of an undergraduate project involving a guided investigation of a more familiar class of algebraic structure, say Boolean algebras or Heyting algebras using these tools.

2 Analysing Larger Proofs

In the second part of this chapter we discuss, using the theory of hoops as a running example, how we have used Prover9 and Mace4 to explore new conjectures, and the methodology we used to analyse and "understand" Prover9's machine generated proofs. The main challenge we want to focus on is in dealing with large Prover9 proofs, and how one should go about breaking these proofs into smaller, more intuitive and understandable steps. As a general methodology, we have adopted the process described in Figure 1.

2.1 A homomorphism property for hoops

As mentioned in Section 1.2, hoops generalise Heyting algebras. Defining the dual of an element as $x^\perp = 1 \ominus x$, we have that in Heyting algebras the double-dual operation $x \mapsto x^{\perp\perp}$ is a homomorphism. The conjecture ϕ we were working on, was whether this

[17] See output file `hp-sum-lemma.txt`.

[18] Prover9 is a theorem-prover for finitely axiomatisable first-order theories: it is not designed to work with something like the principle of induction that can only be expressed either as an infinite axiom schema or as a second-order property. The use of interactive proof assistants that can handle induction is of potential interest in mathematics education, but is not the focus of the present chapter. There has been research on fully automated proof in higher-order logic, but this is in its early days.

1. Start with a new conjecture ϕ
2. Use Mace4 to check ϕ does not have trivial (small) counterexamples
3. User Prover9 to search for a proof of ϕ
   (a) Once proof found, mine the proof for new "concepts" and "properties"
   (b) Rerun the proof search taking these new concepts and properties as given
   (c) Use knowledge learned, formulate new conjecture, and go back to 1.

Figure 1: Methodology

was also the case for hoops, i.e. do the following two homomorphism properties hold:

$$(x \ominus y)^{\perp\perp} = x^{\perp\perp} \ominus y^{\perp\perp} \tag{9}$$

and

$$(x \oplus y)^{\perp\perp} = x^{\perp\perp} \oplus y^{\perp\perp} \tag{10}$$

Using Mace4 we were able to check in just a few minutes that no small (size 20 or below) counter-examples existed[19]. To our surprise, Prover9 found a proof of (9) is just over 100 minutes[20]. This proof, however, is not as short as the ones we have seen in the previous section, involving around 177 steps.

2.2 Discovering derived operations and their basic properties

When faced with a long Prover9 derivation such as the one above, we tried to identify new concepts and intermediate steps in the proof that had intrinsic value, and could be understood in isolation. For instance, we noticed that the patterns $x^{\perp} \oplus (x \ominus y)$ and $x \ominus (x \ominus y)$ appeared multiple times in the derivation. This led us to introduce new operations so that multiple steps in the proof could be understood as properties of these new operations. In total we found, apart from $x \cup y$, three further new derived operations:

$$\begin{aligned} x \cup y &\equiv x \oplus (y \ominus x) \\ x \cap y &\equiv x \ominus (x \ominus y) \\ y \setminus x &\equiv (x \oplus y) \ominus x \\ x \downarrow y &\equiv x^{\perp} \oplus (x \ominus y) \end{aligned}$$

Our final choice of notation for these new operations came after we had studied their properties. The Prover9 symbols we used for these (in ASCII) are shown in Table 4. When identifying these operations we also used our knowledge of the correspondence

[19] See output files `conjectureNNSNNSNN.txt` and `conjecturePNNNNPNN.txt`.
[20] See output file `theoremNNSNNSNN-eq-expanded.txt`.

between hoops and Heyting algebras. For instance, $x \cap y$ in logical terms corresponds to $(y \to x) \to x$, which generalises double negation and in theoretical computer science is known as the *continuation monad* [10].

So, according to step 3. (a) and (b) of our methodology, we looked first for basic properties of these new operations, or of their relation with the primitive operations. We come up with six simple properties (listed in the following lemma) that we then added as axioms, and rerun the proof search.

Lemma 2.1 *The following hold in all hoops:*

(i) $x \geq y \cap x$

(ii) $x \geq x \setminus y$

(iii) $(x \setminus y) \ominus x = 0$

(iv) $x \oplus y = x \oplus (y \setminus x)$

(v) $z \cap (y \ominus x) \geq (z \cap y) \ominus (z \cap x)$

(vi) $x \ominus (x \cap y) = x \ominus y$

Adding these lemmas cut the proof search time to just over 10 minutes[21], and the number of steps to 132. This is still a resonably large proof, which would be hard to "understand" as a whole. So we continued looking for more complex properties of these new defined operations. This time we focused on the following four steps in the proof script `theoremNNSNNSNN-eq-basic-lemmas.txt`, and noticed that these do have intrinsic value:

```
126  (x ~ y) + (1 ~ x) = 1 ~ (x ~ (x ~ y)).
```

states a duality between $x \downarrow y$ and $x \cap y$, i.e. $x \downarrow y = (x \cap y)^{\perp}$.

```
132  (x ~ y) + (1 ~ x) = (y ~ x) + (1 ~ y).
```

states the commutativity of $x \downarrow y$, i.e. $x \downarrow y = y \downarrow x$.

```
134  1 ~ (x ~ (x ~ y)) = 1 ~ (y ~ (y ~ x)).
```

states the commutativity of $x \cap y$ under $(\cdot)^{\perp}$, i.e. $(x \cap y)^{\perp} = (y \cap x)^{\perp}$.

```
153  1 ~ (x ~ (1 ~ (1 ~ x))) = 1.
```

immediately implies that although $x \ominus x^{\perp\perp}$ is not 0 in general, we do have that $(x \ominus x^{\perp\perp})^{\perp\perp} = 0$; and, as we will see, this is the crucial lemma in the proof of (9).

In order to emphasise how we were able to break this long proof into a small collection of simple lemmas (each with a reasonably short proof), we will explicitly give the proof of these lemmas, and the proof of the conjecture from these lemmas. Readers who are

[21] See output file `theoremNNSNNSNN-eq-basic-lemmas.txt`.

Operator	Prover9	Letter	Intuition
0	`0`	Z	(*zero*)
1	`1`	O	(*one*)
$\oplus$	`+`	P	(*plus*)
$\ominus$	`~`	S	(*subtraction*)
$\cup$	`cup`	J	(*join*)
$\cap$	`cap`	M	(*meet*)
$\setminus$	$\setminus$	D	(*difference*)
$\downarrow$	`nand`	A	(*ampheck*)[22]
$(\cdot)^{\perp}$	`(.)'`	N	(*negation*)

Table 4: Nomenclature

not planning to undertake this kind of work themselves are invited to skip the details. We believe, however, that the details will be helpful to those wanting to apply a similar methodology in other contexts.

Notation. But before we do that, let us set up a notation for naming hoop properties. We associate a letter to each of the operations as shown in Table 4 and name each property by reading all the operations on the statement of the property from left to right. For instance, the property $x \downarrow y = y \downarrow x$ is named AA. Although there is a risk that two different properties will end up with the same name, this is not the case for the properties we consider here.

2.3 Discovering basic properties

The first set of basic properties we discovered relate to commutativity. One of the hoop axioms states that $x \cup y$ is commutative. It is easy to construct a model, however, which shows that $x \cap y$ is not commutative in general. When analysing proofs generated by Prover9 we spotted two other interesting commutativity properties, alluded to above. The first (which we call AA as discussed above) is that $x \downarrow y$ also satisfies the commutativity property:

Lemma 2.2 (AA) $x \downarrow y = y \downarrow x$

Proof: By symmetry it is enough to prove $x \downarrow y \geq y \downarrow x$. Note that $((y \ominus x) \ominus x^{\perp}) = 0$, hence:

$$\begin{aligned} (x \ominus y) \oplus x^{\perp} &= (x \ominus y) \oplus x^{\perp} \oplus ((y \ominus x) \ominus x^{\perp}) && \text{Axiom (3)} \\ &= (x \ominus y) \oplus (y \ominus x) \oplus (x^{\perp} \ominus (y \ominus x)) && \text{Axiom (6)} \end{aligned}$$

[22] "Ampheck" from a Greek word meaning "cutting both ways" was the name coined by C.S. Peirce for the logical NAND operation.

$$
\begin{aligned}
&= (x \ominus y) \oplus (y \ominus x) \oplus ((y \ominus x) \oplus x)^{\perp} && \text{Axiom (5)}\\
&= (x \ominus y) \oplus (y \ominus x) \oplus (y \oplus (x \ominus y))^{\perp} && \text{Axiom (6)}\\
&= (y \ominus x) \oplus (x \ominus y) \oplus (y^{\perp} \ominus (x \ominus y)) && \text{Axiom (5)}\\
&= (y \ominus x) \oplus y^{\perp} \oplus ((x \ominus y) \ominus y^{\perp}) && \text{Axiom (6)}\\
&\geq (y \ominus x) \oplus y^{\perp}. && \text{Monotonicity}
\end{aligned}
$$

We also identified this interesting duality between $x \cap y$ and $x \downarrow y$:

Lemma 2.3 (MNA) $(x \cap y)^{\perp} = x \downarrow y$

Proof: By Lemma 2.1 (i) we have $y \geq x \cap y$; and by [EFQ] we have $1 \geq x$. Hence, $(x \cap y)^{\perp} \geq x \ominus y$ so that $(*)$ $(x \ominus y) \ominus (x \cap y)^{\perp} = 0$. Clearly we also have that $(\dagger)$ $(x \ominus y) \ominus x = 0$. Therefore

$$
\begin{aligned}
(x \cap y)^{\perp} &= (x \cap y)^{\perp} \oplus ((x \ominus y) \ominus (x \cap y)^{\perp}) && (*)\\
&= (x \ominus y) \oplus ((x \cap y)^{\perp} \ominus (x \ominus y)) && \text{Axiom (6)}\\
&= (x \ominus y) \oplus ((x \ominus y) \oplus (x \cap y))^{\perp} && \text{Axiom (5)}\\
&= (x \ominus y) \oplus (x \oplus ((x \ominus y) \ominus x))^{\perp} && \text{Axiom (6)}\\
&= (x \ominus y) \oplus x^{\perp}. && (\dagger)
\end{aligned}
$$

This duality when combined with Lemma 2.2 immediately implies that $x \cap y$ is also commutative when in the following weaker form:

Lemma 2.4 (MNMN) $(x \cap y)^{\perp} = (y \cap x)^{\perp}$

Note that we make the point of giving the full proofs of all the lemmas in order to emphasise our goal of reducing the overall proof to a sequence of simple yet interesting lemmas, each of which should have reasonably short proofs (around 10 steps). The other steps in the proof `theoremNNSNNSNN-eq-basic-lemmas.txt` which we found of interest were 101, 138 and 144, which again all have short proofs as follows:

Lemma 2.5 (NPJSSO) $x^{\perp} \oplus ((y \cup x) \ominus (y \ominus x)) = 1$

Proof: Note that $(*)\ x^\perp = (y \ominus x) \oplus (x^\perp \ominus (y \ominus x))$. We have

$$\begin{aligned}
1 &\geq x^\perp \oplus ((y \cup x) \ominus (y \ominus x)) && \text{Axiom (8)}\\
&= x^\perp \oplus ((y \oplus (x \ominus y)) \ominus (y \ominus x)) && \text{Def. } \cup\\
&= (y \ominus x) \oplus (x^\perp \ominus (y \ominus x)) \oplus ((y \oplus (x \ominus y)) \ominus (y \ominus x)) && (*)\\
&= y \oplus (x \ominus y) \oplus (x^\perp \ominus (y \ominus x)) \oplus ((y \ominus x) \ominus (y \oplus (x \ominus y))) && \text{Axiom (6)}\\
&= x \oplus (y \ominus x) \oplus (x^\perp \ominus (y \ominus x)) \oplus ((y \ominus x) \ominus (y \oplus (x \ominus y))) && \text{Axiom (6)}\\
&\geq x \oplus (y \ominus x) \oplus (x^\perp \ominus (y \ominus x)) && \text{Monotonicity}\\
&= x \oplus x^\perp \oplus ((y \ominus x) \ominus x^\perp) && \text{Axiom (6)}\\
&\geq x \oplus x^\perp && \text{Monotonicity}\\
&\geq 1. && \text{Residuation}
\end{aligned}$$

Lemma 2.6 (NSNSM) $x^\perp = (x \ominus y)^\perp \ominus (y \cap x)$

Proof: Note that $(*)\ (x \ominus y) \ominus x = 0$. Hence

$$\begin{aligned}
x^\perp &= (x \oplus ((x \ominus y) \ominus x))^\perp && (*)\\
&= ((x \ominus y) \oplus (x \ominus (x \ominus y)))^\perp && \text{Axiom (6)}\\
&= ((x \ominus y) \oplus (x \cap y))^\perp && \text{Def. } \cap\\
&= (x \cap y)^\perp \ominus (x \ominus y) && \text{Axiom (5)}\\
&= (y \cap x)^\perp \ominus (x \ominus y) && \text{Lemma 2.4}\\
&= (x \ominus y)^\perp \ominus (y \cap x). && \text{Axioms (2) and (5)}
\end{aligned}$$

Lemma 2.7 (NNSSNN) $x^\perp = x^\perp \ominus (x \ominus x^{\perp\perp})$

Proof: It is easy to show that $(*)\ x \oplus (x^\perp \ominus (x \ominus x^{\perp\perp})) = 1$. Let us use the abbreviation $X = x \ominus x^{\perp\perp}$. It is also easy to see that $(\dagger)\ ((X^\perp \ominus x) \ominus ((x \oplus (x^\perp \ominus X)) \ominus x)) = 0$. Hence

$$\begin{aligned}
x^\perp &= 1 \ominus x && \text{Def. } (\cdot)^\perp\\
&= (x \oplus (x^\perp \ominus (x \ominus x^{\perp\perp}))) \ominus x && (*)\\
&= ((x \oplus (x^\perp \ominus (x \ominus x^{\perp\perp}))) \ominus x) \oplus ((X^\perp \ominus x) \ominus ((x \oplus (x^\perp \ominus X)) \ominus x)) && (\dagger)\\
&= ((x \ominus x^{\perp\perp})^\perp \ominus x) \oplus (((x \oplus (x^\perp \ominus X)) \ominus x) \ominus (X^\perp \ominus x)) && \text{Axiom (6)}\\
&= (x \ominus x^{\perp\perp})^\perp \ominus x && \text{Axiom (5)}\\
&= x^\perp \ominus (x \ominus x^{\perp\perp}). && \text{Axiom (5)}
\end{aligned}$$

The above three lemmas are interesting, in the sense that it describes properties of the duality operation $x^\perp$, either showing equivalent ways of writing $x^\perp$, or how it relates to other complex expressions.

Finally, the crucial lemma of the proof shows that, although $x \ominus x^{\perp\perp} \neq 0$ in general, we always have $1 \ominus (x \ominus x^{\perp\perp}) = 1$.

Lemma 2.8 (SNNNO) $(x \ominus x^{\perp\perp})^\perp = 1$

Proof: Note that $(*)\ x^\perp \oplus ((x \ominus x^\perp) \ominus x^{\perp\perp}) = x^\perp$ since $(x \ominus x^\perp) \ominus x^{\perp\perp} = 0$. Hence,

$$
\begin{aligned}
(x \ominus x^{\perp\perp})^\perp &= (x \ominus x^{\perp\perp})^\perp \oplus (x^\perp \ominus x^\perp) && \text{Easy} \\
&= (x \ominus x^{\perp\perp})^\perp \oplus ((x^\perp \oplus ((x \ominus x^\perp) \ominus x^{\perp\perp})) \ominus x^\perp) && (*) \\
&= (x \ominus x^{\perp\perp})^\perp \oplus ((x^\perp \oplus ((x \ominus x^{\perp\perp}) \ominus x^\perp)) \ominus x^\perp) && \text{Axioms (2) and (5)} \\
&= (x \ominus x^{\perp\perp})^\perp \oplus ((x^\perp \cup (x \ominus x^{\perp\perp})) \ominus x^\perp) && \text{Def. } \cup \\
&= (x \ominus x^{\perp\perp})^\perp \oplus ((x^\perp \cup (x \ominus x^{\perp\perp})) \ominus x^\perp) && \text{Def. } \cup \\
&= (x \ominus x^{\perp\perp})^\perp \oplus ((x^\perp \cup (x \ominus x^{\perp\perp})) \ominus (x^\perp \ominus (x \ominus x^{\perp\perp}))) && \text{Lemma 2.7} \\
&= 1. && \text{Lemma 2.5}
\end{aligned}
$$

Lemma 2.8 immediately implies step 159 of `theoremNNSNNSNN-eq-basic-lemmas.txt`, namely:

Lemma 2.9 (SSNNSNO) $((x \ominus y) \ominus (x^{\perp\perp} \ominus y))^\perp = 1$

Proof: We have

$$
\begin{aligned}
1 &= (x \ominus x^{\perp\perp})^\perp && \text{Lemma 2.8} \\
&\leq ((x \ominus x^{\perp\perp}) \ominus (y \ominus x^{\perp\perp}))^\perp && \text{Monotonicity} \\
&= (x \ominus (x^{\perp\perp} \oplus (y \ominus x^{\perp\perp})))^\perp && \text{Axiom (5)} \\
&= (x \ominus (y \oplus (x^{\perp\perp} \ominus y)))^\perp && \text{Axiom (6)} \\
&= ((x \ominus y) \ominus (x^{\perp\perp} \ominus y))^\perp && \text{Axiom (5)} \\
&\leq 1
\end{aligned}
$$

2.4 Producing a human-readable proof of (9)

We are now in a position where we can derive a human-readable proof of the homomorphism property (9) using the lemmas of the previous section. Our proof is based on the one in the proof script `theoremNNSNNSNN-eq-expanded.txt`.

Theorem 2.10 (NNSNNSNN) $x^{\perp\perp} \ominus y^{\perp\perp} = (x \ominus y)^{\perp\perp}$

Proof: Since $(x^{\perp\perp} \ominus y) \ominus (x \ominus y) = 0$ it follows that $(*)$ $(x^{\perp\perp} \ominus y) \cap (x \ominus y) = x^{\perp\perp} \ominus y$. Hence

$$\begin{aligned}
x^{\perp\perp} \ominus y^{\perp\perp} &= (x^{\perp\perp} \ominus y)^{\perp\perp} && \text{Lemma 2.1 } (vi)\\
&= (1 \ominus (x^{\perp\perp} \ominus y))^{\perp} && \text{Def. } (\cdot)^{\perp}\\
&= (1 \ominus ((x^{\perp\perp} \ominus y) \cap (x \ominus y)))^{\perp} && (*)\\
&= (((x \ominus y) \ominus (x^{\perp\perp} \ominus y))^{\perp} \ominus ((x^{\perp\perp} \ominus y) \cap (x \ominus y)))^{\perp} && \text{Lemma 2.9}\\
&= (x \ominus y)^{\perp\perp}. && \text{Lemma 2.6}
\end{aligned}$$

2.5 Tackling the harder conjecture (10)

We have also been able to produce a human-readable proof of (10), although this seems to be a much harder result. Prover9 is also able to find a proof of (10) from the basic hoop axioms, but that takes almost 7 hours, and requires 624 steps[23]. In a process of "mining" this proof for new lemmas, and then searching for a proof again using these lemmas, we were able to find a small set of lemmas from which Prover9 derives the proof of the conjecture in just a fraction of a second[24]. Again, in order to emphasise how we were able to produce a human-like mathematical presentation of this proof, we prove here these other lemmas in full, and give the (short) proof of (10) from these lemmas.

The first two lemmas enable us to rewrite x or $x^{\perp}$ as a sum of two other elements. In a hoop we do not have idempotence ($x = x \oplus x$) in general, but we can obtain weaker forms of this as follows:

Lemma 2.11 (MPS) $x = (x \cap y) \oplus (x \ominus y)$

Proof: We have

$$\begin{aligned}
x &= x \oplus ((x \ominus y) \ominus x) && \text{Axiom (3)}\\
&= (x \ominus (x \ominus y)) \oplus (x \ominus y) && \text{Axiom (6)}\\
&= (x \cap y) \oplus (x \ominus y). && \text{Def. } \cap
\end{aligned}$$

Lemma 2.12 (NSPJN) $x^{\perp} = (y \ominus x) \oplus (x \cup y)^{\perp}$

Proof: That $x^{\perp} \geq (y \ominus x) \oplus (x \cup y)^{\perp}$ follows directly since $(x \cup y)^{\perp} = (y \ominus x)^{\perp} \ominus x$. For the other direction we have:

$$\begin{aligned}
x^{\perp} &= x^{\perp} \oplus (y \ominus 1) && \text{Axiom (8)}\\
&\geq x^{\perp} \oplus ((y \ominus x) \ominus x^{\perp}) && \text{Easy}\\
&= (y \ominus x) \oplus (x^{\perp} \ominus (y \ominus x)) && \text{Axiom (6)}\\
&= (y \ominus x) \oplus (x \oplus (y \ominus x))^{\perp} && \text{Axiom (5)}\\
&= (y \ominus x) \oplus (x \cup y)^{\perp}. && \text{Def. } \cup
\end{aligned}$$

[23] See file `theoremPNNNNPNN-eq.txt`.
[24] See file `theoremPNNNNPNN-eq-lemmas.txt`.

The following two lemmas can be seen as properties relating $x \oplus y$ with the new derived connectives $x \cup y$, $x \cap y$ and $x \setminus y$.

Lemma 2.13 (PPMD) $x \oplus y = x \oplus (y \cap (y \setminus x))$

Proof: Note that $(*)$ $(y \ominus (y \setminus x)) \ominus x = 0$. Hence

$$\begin{aligned}
x \oplus y &= (y \setminus x) \oplus x && \text{Lemma 2.1 } (iv)\\
&= (y \setminus x) \oplus x \oplus ((y \ominus (y \setminus x)) \ominus x) && (*)\\
&= (y \setminus x) \oplus (y \ominus (y \setminus x)) \oplus (x \ominus (y \ominus (y \setminus x))) && \text{Axiom (6)}\\
&= y \oplus ((y \setminus x) \ominus y) \oplus (x \ominus (y \ominus (y \setminus x))) && \text{Axiom (6)}\\
&= y \oplus (x \ominus (y \ominus (y \setminus x))) && \text{Lemma 2.1 } (iii)\\
&= (y \cap (y \setminus x)) \oplus (y \ominus (y \setminus x)) \oplus (x \ominus (y \ominus (y \setminus x))) && \text{Lemma 2.11}\\
&= (y \cap (y \setminus x)) \oplus x \oplus ((y \ominus (y \setminus x)) \ominus x) && \text{Axiom (6)}\\
&= x \oplus (y \cap (y \setminus x)). && \text{Monotonicity}
\end{aligned}$$

Lemma 2.14 (NPNPM) $x^\perp \oplus y = x^\perp \oplus (y \cap x)$

Proof: That $x^\perp \oplus y \geq x^\perp \oplus (y \cap x)$ follows directly from $y \geq y \cap x$. For the other direction, note that $x \geq y \setminus x^\perp$. Hence, $y \cap x \geq y \cap (y \setminus x^\perp)$. Therefore, the result follows directly from Lemma 2.13.

The above lemmas give us another interesting and useful duality between the two defined operations $x \cap y$ and $x \setminus y$:

Lemma 2.15 (JNND) $(x \cup y)^\perp = y^\perp \setminus x$

Proof: That $y^\perp \setminus x \geq (x \cup y)^\perp$ is easy to show. For the converse, observe that by Lemma 2.14 (with y and x interchanged) it follows that $(*)$ $y^\perp \geq (x + y^\perp) \ominus (x \cap y)$. Hence

$$\begin{aligned}
(x \cup y)^\perp &= (x \oplus (y \ominus x))^\perp && \text{Def. } \cup\\
&= (y \oplus (x \ominus y))^\perp && \text{Axiom (6)}\\
&= y^\perp \ominus (x \ominus y) && \text{Axiom (5)}\\
&\geq ((x \oplus y^\perp) \ominus (x \cap y)) \ominus (x \ominus y) && (*)\\
&= ((x \oplus y^\perp) \ominus ((x \ominus y) \ominus x)) \ominus x && \text{Axiom (6)}\\
&= (x \oplus y^\perp) \ominus x && \text{Easy}\\
&= y^\perp \setminus x. && \text{Def. } \setminus
\end{aligned}$$

The above, together with Lemma 2.15, immediately gives us another interesting commutativity property.

Corollary 2.16 (NDND) $y^\perp \setminus x = x^\perp \setminus y$

The final main lemma in the proof of (10) is a surprising duality between $\ominus$ and $\oplus$.

Lemma 2.17 (SNNNPN) $(y \ominus x^\perp)^\perp = x^\perp \oplus y^\perp$

Proof: That $x^\perp \oplus y^\perp \geq (y \ominus x^\perp)^\perp$ is easy to derive. For the converse, note that we have $x^{\perp\perp} \geq y \ominus x^\perp$, and hence

$$(*)\ (x^{\perp\perp} \oplus (y \ominus x^\perp)^\perp) \ominus x^{\perp\perp} \geq (x^{\perp\perp} \oplus x^{\perp\perp\perp}) \ominus x^{\perp\perp} = x^{\perp\perp\perp}.$$

Hence, taking $x' = y \ominus x^\perp$ and $y' = x^{\perp\perp}$ in Lemma 2.12, we have the first line of the following chain

$$\begin{aligned}
(y \ominus x^\perp)^\perp &= (x^{\perp\perp} \ominus (y \ominus x^\perp)) \oplus ((y \ominus x^\perp) \cup x^{\perp\perp})^\perp && \text{Lemma 2.12}\\
&= (x^{\perp\perp} \ominus (y \ominus x^\perp)) \oplus ((y \ominus x^\perp)^\perp \setminus x^{\perp\perp}) && \text{Theorem 2.15}\\
&\geq (x^{\perp\perp} \ominus (y \ominus x^\perp)) \oplus x^{\perp\perp\perp} && (*)\\
&= (x^\perp \oplus (y \ominus x^\perp))^\perp \oplus x^\perp && \text{Lemma 2.1 } (vi)\\
&= (x^\perp \cup y)^\perp \oplus x^\perp && \text{Def } \cup\\
&= (x^{\perp\perp} \setminus y) \oplus x^\perp && \text{Theorem 2.15}\\
&= (y^\perp \setminus x^\perp) \oplus x^\perp && \text{Corollary 2.16 } (i)\\
&\geq x^\perp \oplus y^\perp. && \text{Residuation}
\end{aligned}$$

Lemma 2.17 above, immediately implies the homomorphism property for $x \oplus y$, since $(x \oplus y)^{\perp\perp} = (y^\perp \ominus x^{\perp\perp})^\perp$.

Theorem 2.18 (PNNNNPNN) $(x \oplus y)^{\perp\perp} = x^{\perp\perp} \oplus y^{\perp\perp}$

Remark 2.1 *It is interesting to observe that in the proof* `theoremPNNNNPNN-eq.txt` *of property (10) one also finds some of the lemmas used in the proof of (9), for instance, Lemmas 2.3 (step 329) and 2.12 (step 486), but more interestingly, it also discovers Lemma 2.17 (step 633), and uses that to derive a more general duality between* $\ominus$ *and* $\oplus$ *(step 683), namely*

$$(x \ominus y)^\perp = x^\perp \oplus y^{\perp\perp}$$

an interesting property, as in the absence of idempotence ($x = x \oplus x$)*, i.e., in a hoop that is not a Heyting algebra, it is usually hard to find non-trivial equivalences between non-sums and sums.*

3 Concluding Remarks

In Section 1 we have attempted to introduce the tools and methods we have been using by examples at the level of an undergraduate project. We hope this is of interest to educators and advocate introduction of tools such as Prover9 and Mace4 into mathematical curricula.

At a more advanced level, we have discussed our own research using Prover9 and Mace4 to investigate algebraic structures. It is possible to demonstrate the provability of properties like duality, commutativity or homomorphism properties by model-theoretic methods but these methods are not constructive, whereas the methods discussed in Section 2 construct explicit equational proofs.

In [6], de Villiers argues that proof has many purposes apart from verification, including explanation, systematization, intellectual challenge, discovery and communication. Tools such as Prover9 automate the process of discovering a proof, but at first glance, the proofs that are discovered seem inacessible to a human reader. We take this as an intellectual challenge in its own right and claim that with human effort, judiciously applied, we can "mine" explanative and systematic human-oriented proofs from machine-generated ones, potentially leading to new insights into the problem domain.

We have deliberately avoiding discussing potential developments of the tools in the main body of this chapter. However, there are several obvious areas for future investigation. Some automated support for refactoring the machine-generated proofs could be very helpful. The refactoring steps of interest would include separating out lemmas and retrofitting derived notations. It is certainly of interest to speculate on possibilities for fully automating extraction of human-readable proofs from machine-generated proofs, but we view this as a hard challenge for Artificial Intelligence.

References

[1] W. J. Blok and I. M. A. Ferreirim. On the structure of hoops. *Algebra Universalis*, 43(2-3):233–257, 2000. 5

[2] B. Bosbach. Komplementäre Halbgruppen. Axiomatik und Arithmetik. *Fundam. Math.*, 64:257–287, 1969. 5, 6

[3] J. R. Büchi and T. M. Owens. Complemented monoids and hoops. Unpublished manuscript, c. 1974. 5, 6

[4] Stanley. Burris. An Anthropomorphized Version of McCune's Machine Proof that Robbins Algebras are Boolean Algebras. Private communication., c. 1997. 1

[5] Bernd I. Dahn. Robbins algebras are Boolean: A revision of McCune's computer-generated solution of Robbins problem. *J. Algebra*, 208(2):526–532, 1998. 1

[6] Michael de Villiers. The role and function of proof in mathematics. *Pythagoras*, 24:17–24, 11 1990. 20

[7] J. Łukasiewicz and A. Tarski. Untersuchungen über den Aussagenkalkül. *C. R. Soc. Sc. Varsovie 23*, (1930):30–50, 1930. 7

[8] W. McCune. Prover9 and Mace4. http://www.cs.unm.edu/~mccune/prover9/, 2005–2010. 1

[9] William McCune. Solution of the Robbins problem. *J. Autom. Reasoning*, 19(3):263–276, 1997. 1

[10] E. Moggi. Computational lambda-calculus and monads. In *Symposium of Logic in Computer Science*, California, June 1989. IEEE. 12

[11] S. Winker. Absorption and idempotency criteria for a problem in near-Boolean algebras. *J. Algebra*, 153(2):414–423, 1992. 1