

Automated Theorem Discovery: A Future Direction for Theorem Provers

Simon Colton

Division of Informatics
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
United Kingdom
simonco@dai.ed.ac.uk

Abstract. One obvious and important aspect of automated theorem proving is that the users know in advance which theorem they wish to prove. A possible future direction for theorem provers is to enable users to discover theorems which they were not necessarily aware of. We survey previous attempts at this and give a new demonstration of theorem generation using our HR program [10] in the domain of ‘anti-associative’ algebras. We also suggest three applications where this functionality may prove useful, and discuss how this would add value to theorem provers.

1 Introduction

The Theorema system [4] is a theorem proving program which is written on top of the Mathematica computer algebra package [26]. Bruno Buchberger has suggested in [3] that users of Theorema explore new domains in a very systematic way. We believe that such a systematic exploration can be automated by an additional module added to theorem provers. We believe this would add much value to automated theorem proving (in terms of attracting interest from mathematicians and the wider scientific community) and that this is a worthy future direction for automated theorem proving.

Automated scientific discovery is emerging as an important sub-field of Artificial Intelligence. Via automated techniques, new and interesting results are being found in many areas of science, including chemistry, biology, medicine, astronomy and particle physics (see [16] for a survey). In mathematics, new results are found routinely using computer algebra systems to perform calculations and algebraic manipulations. In these cases, the nature and impact (in terms of implication and application) of the result is largely known in advance, and the computer only fills in the value of the calculation to facilitate the statement of the final result. Occasionally, the value of the calculation is surprising, which leads to further conjectures, advancing knowledge of the domain. However, it is fair to say that computer algebra systems are rarely seen as ‘discovering’ a result which was fundamentally unknown and surprising to the user.

Theorem proving techniques have occasionally proved theorems which were previously unsolved, in particular the Robbins Algebra conjecture [21]. Again, the nature and impact of the result were known in advance, and the only surprise was that the theorem is actually true. It is more acceptable to project the act of ‘discovery’ onto theorem provers than computer algebra systems. This is because a certain amount of search is involved in finding often complicated proofs. We are interested in extending the nature of the discovery tasks to *finding* as well as proving interesting theorems in a given domain of interest. We believe that an ability to find interesting theorems which the user would not necessarily think of would add much value to theorem provers.

1.1 Background

There have been a number of projects where automated techniques have discovered new conjectures¹ but few where the conjectures have been proved automatically as well. We know of only three applications, other than the one described in §2 below, where theorem proving has been used to discover new theorems rather than a proof to an existing conjecture.

McCune and Padmanabhan have used exhaustive searches with automated theorem provers such as Otter [17] to find single axioms for algebras including groups, Abelian groups and ternary boolean algebras [18], [19], [22]. The exhaustive search was very restrictive and the nature and impact of the final result was known in advance. However, the results were surprising and important because they identified smaller axiomatisations than were perhaps expected.

In [28] Zhang describes the MCS model based conjecture searching program. MCS used a five step process whereby:

- (i) models of a finite algebra were found,
- (ii) a set of closed well-formed formulae was generated
- (iii) the formulae were split into three sets: S_0 : false for all the models, S_1 : true for all the models and S_2 true for some, but not all of the models
- (iv) conjectures in S_0 were discarded and machine learning techniques were employed to find relationships between formulae in S_2
- (v) the truth of the formulae in S_1 was proved (or disproved using more model generation to find a conjecture).

This approach was successful, finding some results in QG5-quasigroup theory which the author stated were generally believed to be helpful in solving the QG5 problem. The search was exhaustive over the set of formulae and again, the format of the results could be predicted in advance as the formulae were simple compositions of the multiplication operation. Zhang stated that combinatorial explosion was a major obstacle to overcome, and that future work would involve

¹ See [2], [5], [11] and [14].

selection of more complex, more interesting conjectures to (attempt to) prove. To our knowledge, this feature has yet to be added to MCS.

The program developed by Rajiv Bagai et al. worked in plane geometry and aimed to find theorems stating that certain idealised diagrams could not be drawn [1]. Starting with an a blank diagram represented in a first order language, constructions such as parallelograms were made by adding new points and lines and new relations (such as two lines being parallel). Each time a new diagram was constructed by adding a relation, a conjecture was made that the diagram was inconsistent, i.e. that it was not possible to draw an example of the diagram. An attempt was then made to prove the conjecture using an efficient theorem prover [6] based on Wu's method [27] and proved conjectures were output as theorems. Certain techniques were employed to cut down on the use of the theorem prover, in particular maintaining information about which diagrams were isomorphic. Not only did the program re-discover well known results such as Euclid's 5th postulate, but Chou also used Wu's method to provide a systematic way to generate and prove possibly new results in geometry [7].

1.2 The HR Program

Our approach to discovery in mathematics has been based upon our HR program [9], [10]. HR has been developed to form theories in domains of pure mathematics starting from minimal information such as the axioms of a finite algebra. HR is a complex program and has been described many times, so we only give a few relevant details here, as the thrust of this paper is not *how* HR performs theorem discovery, but rather that doing so is a good idea.

HR forms a theory by repeatedly performing theory formation steps. Steps start with an effort to form a new concept from an old one via one of seven general production rule steps. This may lead to a new concept, but equally it may lead to the re-invention of an old concept. HR checks whether a concept is new by checking that it has different examples to all previous concepts. If a match is found, HR conjectures that the new concept and the old one are logically equivalent and this conjecture is passed to the Otter theorem prover [17]. If Otter fails to prove it, the conjecture is passed to the MACE model generator [20] which attempts to find a counterexample. In order to give Otter every chance to prove the theorem, prime implicates are extracted from the equivalence conjectures, where prime implicates are implication conjectures where no subset of the premises implies the goal. HR also makes non-existence conjectures by noticing that a newly formed concept has no examples, and it finds implication conjectures by noticing that one concept subsumes another.

Information from the concepts, conjectures, theorems and proofs is used to assess the interestingness of the concepts, so that a heuristic search can be undertaken whereby the most interesting concepts are used in theory formation steps before the less interesting ones. For instance, HR uses measures of the complexity of definitions, parsimony of examples and novelty of the categorisations achieved, as discussed in [12]. In addition, the conjectures that a concept appears in are also assessed to give another measure for the worth of the concept. That

is, concepts appearing in interesting conjectures are considered more interesting than those which do not. Equivalence conjectures are assessed in terms of their surprisingness, i.e. how different the left hand and right hand concepts are, and all conjectures which are proved by Otter are assessed in terms of the length of the proof which Otter found (a statistic supplied in Otter's output). An overall value for concepts is evaluated using a weighted sum over all the measures, with the weights set by the user.

This approach is general enough for HR to have been used successfully in around 20 finite algebras, including group, quasigroup and ring theory, as well as graph theory and number theory (although due to the numerical nature of the last two domains, no theorem proving was possible). Using an additional module which enabled HR to datamine the Encyclopedia of Integer Sequences [23] HR has invented many new sequences of integers and provided evidence that they were interesting enough to be accepted into the Encyclopedia. For instance, HR invented the concept of integers with a prime number of divisors and also conjectured that if the sum of divisors of an integer is prime, then the number of divisors will also be prime (a result we later proved). Details of the application to number theory are given in [8] and [11].

2 Worked Example - Anti-Associative Algebras

In [9], we wished to show that HR could be used not only for making *conjectures* (as it did with the integer sequences), but also to discover *theorems*, by presenting only those conjectures which it had proved with Otter.

We wanted to use HR in a domain which was completely new to us and see if the theory it produced held any surprising results. We decided to use HR to explore 'anti-associative' algebras which have only one axiom: that no triple of elements are associative, i.e. $\forall a, b, c (a * b) * c \neq a * (b * c)$. We are not aware of any work in this domain, which is different to non-associative algebras, where the only condition is that *one* triple of elements is not associative. We ran HR with the default algebra settings for 1000 theory formation steps. We first noticed that HR had used MACE to find 34 examples of anti-associative algebras from size 2 to 6, including these two of size 6:

*	0	1	2	3	4	5		*	0	1	2	3	4	5
0	1	5	5	1	0	1		0	1	3	5	0	5	2
1	4	2	2	4	2	2		1	4	2	3	2	0	2
2	3	3	0	3	0	3		2	3	1	0	0	0	0
3	4	2	2	4	2	2		3	3	4	5	0	5	1
4	1	5	5	1	5	1		4	3	1	3	0	0	0
5	3	3	0	3	0	3		5	3	5	2	0	0	0

As it was not obvious to us that there would be any examples of this type of algebra, it was interesting that relatively large examples existed. Conversely, it was also interesting that examples of size two existed, because the anti-associative axiom appears to be fairly constraining. These two isomorphic examples were found by HR:

$$\begin{array}{c|c} * & 0 & 1 \\ \hline 0 & 1 & 1 \\ 1 & 0 & 0 \end{array} \qquad \begin{array}{c|c} * & 0 & 1 \\ \hline 0 & 1 & 0 \\ 1 & 1 & 0 \end{array}$$

There were no examples of anti-associative algebras of size 1 or 5. In the first case, it is easy to see that the trivial algebra cannot have the anti-associative property and HR actually conjectures and proves this. We have subsequently used MACE to find an example of size five and we now know that there are examples of all sizes greater than 1: multiplication tables where the first column contains all ones and the other columns all contain zeros have the anti-associativity property. For example, this multiplication table of size 5 is anti-associative:

$$\begin{array}{c|cccc} * & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 0 & 0 \end{array}$$

We conjectured that all multiplication tables of this type are anti-associative after looking at the examples found by HR (HR did not provide the conjecture explicitly). The conjecture is true, proved by the following case split:

If $c = 0$ then $\forall a, b (a * b) * c = 1$, but $(b * c) = 1$, so $a * (b * c) = 0 \neq (a * b) * c$.
If $c \neq 0$ then $\forall a, b (a * b) * c = 0$, but $(b * c) = 0$, so $a * (b * c) = 1 \neq (a * b) * c$.

To further investigate the theory, we wished to find out what properties the complete lack of associativity rules out. For example, it is obvious that groups (which are associative) do not have the anti-associative property and we wanted to find some results of this nature which were less obvious. There were 240 theorems in the theory (all proved by Otter) and we listed them in terms of decreasing proof length, so that we could examine the most difficult (in Otter's terms) first. We first observed theorem 168:

$$\nexists a \text{ s.t. } \forall b (b * b = a)$$

This states that there must be at least two different elements on the diagonal line of the multiplication table. This was not obvious to us, as it is not true of many other algebraic domains, for instance quasigroups, groups, etc.

Next, we noticed theorem 154:

$$\nexists a, b ((\exists c \text{ s.t. } a * c = b) \& (\exists d \text{ s.t. } d * a = b))$$

This states that these algebras cannot be quasigroups. Again, this was certainly not obvious to us, and a corollary of this is interesting: at least one triple of elements in quasigroups must be associative. Furthermore, theorem 47 was a stronger result about the non-quasigroup nature of anti-associative algebras:

$$\nexists a \text{ s.t. } (\forall c (\exists d (c * d = a) \& \exists e (e * a = c)))$$

This states that if the n th row has all the elements in it, then the n th column will *not* have all the elements in it (and vice-versa).

Following this, we noticed theorem 12:

$$\nexists a \text{ s.t. } a * a = a,$$

This states that there are no idempotent elements. This was not surprising because if an element a is idempotent, then the triple (a, a, a) will be associative.

We also noticed theorems 138 and 139:

$$\nexists a \text{ s.t. } \forall c (a * c = c)$$

$$\nexists a \text{ s.t. } \forall c (c * a = c)$$

These state that there can be no global left or right identities. Therefore, there can be no identity element, as there is in a group. This was also not surprising, because an identity element would be idempotent. However, upon looking at the prime implicates that HR found, we noticed a stronger condition about identities which was not obvious:

$$\forall a, b \ a * b = a \Rightarrow b * a \neq a$$

This states that if b is a right identity of a , then it cannot be a left identity of a , hence no element has a local identity.

Furthermore, we noticed two slightly surprising prime implicates in HR's theory:

$$\forall a, b, c \ (a * a = b \ \& \ c * c = a \Rightarrow c * c \neq b) \quad (1)$$

$$\forall a, b, c \ (a * b = c \ \& \ b * a = c \Rightarrow a * a \neq b) \quad (2)$$

After a little rearranging of 1, we can state it as: $\forall a, (a*a)*(a*a) \neq (a*a)$ and we see that it is just a special case of the theorem that no element is idempotent, so it is less surprising than we thought. We have included (1) here as an indication of the uninteresting results HR produced. In contrast, theorem 2 shows that if two elements commute, neither will be the square of the other, which was not obvious.

Disappointingly, in this session, HR did not invent the concept of Abelianness. We noticed that HR initially found the concept of commutative pairs interesting and developed it by combining it with other concepts. However, after this initial development, the interestingness rating dropped because the theorems produced were relatively easy to prove, so commutative pairs were not developed again. After the session, we used HR to explore the domain by ourselves, by forcing particular theory formation steps. When we attempted to invent the concept of Abelian anti-associative algebras, HR made and proved the conjecture that none exist. Finally we tried to get HR to invent the concept of central elements (i.e. those which commute with all the others). Again, HR proved that no such elements exist in anti-associative algebras, hence they cannot even have a non-empty centre. While HR did not tell us this directly, it was very easy to use HR to explore the theory ourselves.

To summarise the main findings in this session, HR showed us that anti-associative algebras cannot be quasigroups and they cannot be Abelian or idempotent or have an identity element. HR also made stronger conjectures about the non-quasigroup, non-Abelian, non-idempotent and non-identity nature of these algebras. HR also highlighted some properties which may help identify algebras of this type, for example that there must be at least two different elements on the diagonal of the multiplication table and that if two elements commute, neither will be the square of the other. We also discovered that there are examples of this algebra of every size greater than 1 (but not one of size 1). We regard all these facts as interesting and all of them were unknown to us before the session. We hope to have shown that HR can be used to discover interesting theorems in finite algebraic domains, and hope this indicates the potential value that discovery tasks add to theorem provers, a point we expand on in the next section.

3 Three Possible Applications

The long-term goal of this project is to advance theorem discovery techniques to the stage where theory formation programs such as HR are as integral a tool for mathematicians as computer algebra systems. However, this goal is a long way off and there are many potential applications of this work along the way, three of which are discussed below.

Firstly, we have had some initial results in applying HR to discovering additional constraints for constraint satisfaction problems (CSPs). As discussed in [13], correct modelling of a CSP is vital to the chances of solving the problem. In particular, adding constraints to CSPs can often greatly reduce the time taken to find solutions. However, the addition of constraints is, for the large part, done by hand. We have applied HR to discovering theorems in domains associated with certain CSPs, with the theorems interpreted as additional constraints for the CSP. For example, working in QG3-quasigroup theory, HR discovered and used Otter to prove that the leading diagonal must be pairwise distinct. The application to CSPs is very preliminary, but the results are encouraging. For instance, adding the above constraint to the QG3-quasigroup CSP produced a 3-fold reduction in the time taken to find an example of size 8. The two main attractions to this approach were that HR found theorems which we would possibly not have thought of, and that these results were presented in an already proved form. Users of constraint satisfaction solvers are not necessarily skilled in theorem proving, so it is important to know that the theorems have been proved, although the exact nature of the proof is largely immaterial.

Secondly, the TPTP database — “a thousand problems for theorem provers” — contains over 4000 theorems and is used as a testbed for automated theorem provers. Furthermore, the CASC competition [24] draws from the TPTP in order to decide which theorem provers are fastest over a set of theorems. We have been encouraged by Geoff Sutcliffe, who maintains the TPTP, to use HR to find new theorems for the TPTP, and we have done some preliminary work towards

this. We must report that, as yet, none of HR's theorems have been of sufficient difficulty to be used in the TPTP. This is because we have mostly used HR to find *simple* theorems in a domain (for instance, with the anti-associative algebras, we found simple, yet interesting results). HR can certainly produce very complicated theorems, but we normally avoid them because they are too difficult for us to understand. However, it is these kinds of theorems which should be encouraged for the TPTP, and we need to improve how HR presents the theorems in order for us to understand them. We are confident that HR will succeed in finding theorems suitable for the TPTP database. Sutcliffe is particularly interested in HR finding theorems in domains which are not yet represented in the database, for example anti-associative algebras.

Finally, we believe that programs such as HR could be used as a mathematics teaching tool. Some of the conjectures HR found in number theory were of a similar complexity and nature as those found as exercises in number theory books, for instance the conjecture that if the sum of divisors of an integer is prime, then the number of divisors will be prime is a good example of this. If HR could find and prove theorems of a similar complexity, then this might be a useful tool for mathematics tutors, i.e. as a source of exercises for students to prove. The attraction of the theorems would be that they have been proved by a theorem prover, so they must be correct² and they will probably be fairly easy, but not necessarily trivial, to prove.

4 Conclusions and Further Work

Much further work is required to make automated theorem provers more attractive to the mathematics community. The automated deduction community is slowly beginning to realise the potential of discovery tasks to add value to automated theorem provers, for instance see [28]. We have not yet suggested general ways in which discovery functionality could be added to theorem provers in general. Rather, we have tried to motivate this direction for theorem provers by demonstrating that current theory forming techniques are available to highlight theorems in a domain which the users would not necessarily have thought of themselves. In particular, we showed how HR used Otter to highlight some surprising results in anti-associative algebra (for instance, that there must be at least two different elements on the diagonal of the multiplication tables of these algebras). We tentatively claim that these results were more surprising in some way than those found by McCune and Padmanabhan, Zhang and Chou's approaches because the search was less restrictive. We have also suggested three possible applications of discovery-enhanced theorem proving and hope that these will provide fruitful new areas in which to apply HR.

There are many future directions for automated theorem discovery research. In particular, we intend to link HR with many theorem provers, either through

² Subject to bugs in the theorem prover's implementation and the way in which HR passes them to the prover.

the MATHWEB [15] or SystemOnTPTP [25] systems (or indeed both). The advantage to HR having a selection of provers to employ is that appropriate ones can be selected for a particular theorem. SystemOnTPTP has such a selection mechanism, and we hope to take full advantage of this. We also intend to extend HR's conjecture making via enhanced data-mining techniques. There are many large databases of mathematical knowledge being developed and our work with one of them — the Encyclopedia of Integer Sequences — has shown that even fairly simple datamining techniques can lead to interesting new conjectures. We intend to improve these techniques and generalise them to work on many databases.

Through integration of discovery techniques with theorem provers, we believe we could add value and possibly make automated theorem proving more attractive to mathematicians and scientists in general. By embedding theory formation programs in computer algebra systems, we also believe that theorem discovery could be a widely used and very valuable tool for mathematicians. Putting this long term goal aside, theorem discovery is certainly a very worthy future direction for automated theorem proving which has many applications in the short term.

5 Acknowledgements

This research is supported by EPSRC research grant GR/M98012. The author is also affiliated with the Department of Computer Science at the University of York. We would like to thank Geoff Sutcliffe for insights into the TPTP database and the SystemOnTPTP, and for providing a new challenge for HR. We would also like to thank the anonymous reviewers for their useful comments on this paper.

References

1. R Bagai, V Shanbhogue, J Żytkow, and S Chou. Automatic theorem generation in plane geometry. In *LNAI 689*, pages 415–424. Springer Verlag, 1993.
2. D Bailey, M Borwein, P Borwein, and S Plouffe. The quest for pi. *Mathematical Intelligencer*, 19(1):50–57, 1997.
3. B Buchberger. Theory exploration versus theorem proving. In *Proceedings of Calculemus 99, Systems for Integrated Computation and Deduction*, 1998.
4. Bruno Buchberger, Tudor Jebelean, Franz Kriftner, Mircea Marin, Elena Tomuta, and Daniela Vasaru. A survey of the theorema project. In *International Symposium on Symbolic and Algebraic Computation*, pages 384–391, 1997.
5. G Caporossi and P Hansen. Finding relations in polynomial time. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 780–785, 1999.
6. S Chou. *Mechanical Theorem Proving*. D. Reidel Publishing Company, Dordrecht, Netherlands, 1984.
7. S Chou. Proving and discovering geometry theorems using Wu's method. Technical Report 49, Computing Science, University of Austin at Texas, 1985.

8. S Colton. Refactorable numbers - a machine invention. *Journal of Integer Sequences*, www.research.att.com/~njas/sequences/JIS, 2, 1999.
9. S Colton. *Automated Theory Formation in Pure Mathematics*. PhD thesis, Division of Informatics, University of Edinburgh, 2001.
10. S Colton, A Bundy, and T Walsh. HR: Automatic concept formation in pure mathematics. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 786–791, 1999.
11. S Colton, A Bundy, and T Walsh. Automatic invention of integer sequences. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 558–563, 2000.
12. S Colton, A Bundy, and T Walsh. On the notion of interestingness in automated mathematical discovery. *International Journal of Human Computer Studies*, 53(3):351–375, 2000.
13. S Colton, L Drake, A Frisch, I Miguel, and T Walsh. Automated generation of implied constraints: Initial progress. In *Proceedings of the Automated Reasoning Workshop*, 2001.
14. S Fajtlowicz. On conjectures of Graffiti. *Discrete Mathematics* 72, 23:113–118, 1988.
15. A Franke, S Hess, C Jung, M Kohlhase, and V Sorge. Agent-oriented integration of distributed mathematical services. *Journal of Universal Computer Science*, 5(3):156–187, 1999. Special Issue on Integration of Deduction Systems.
16. P Langley. The computer-aided discovery of scientific knowledge. In *Proceedings of the first international conference on discovery science*, 1998.
17. W McCune. The OTTER user's guide. Technical Report ANL/90/9, Argonne National Laboratories, 1990.
18. W McCune. Automated discovery of new axiomatizations of the left group and right group calculi. *Journal of Automated Reasoning*, 9(1):1–24, 1992.
19. W McCune. Single axioms for groups and abelian groups with various operations. *Journal of Automated Reasoning*, 10(1):1–13, 1993.
20. W McCune. A Davis-Putnam program and its application to finite first-order model search. Technical Report ANL/MCS-TM-194, Argonne National Laboratories, 1994.
21. W McCune. Solution of the Robbins problem. *Journal of Automated Reasoning*, 19(3):263–276, 1997.
22. R Padmanabhan and W McCune. Single identities for ternary boolean algebras. *Computers and Mathematics with Applications*, 29(2):13–16, 1995.
23. N Sloane. The Online Encyclopedia of Integer Sequences. <http://www.research.att.com/~njas/sequences>, 2000.
24. G Sutcliffe. The CADE-16 ATP system competition. *Journal of Automated Reasoning*, 24(3):371–396, 2000.
25. G Sutcliffe. SystemOnTPTP. In *Proceedings of CADE-17, LNAI, 1831*. Springer Verlag, 2000.
26. S Wolfram. *The Mathematica Book, Fourth Edition*. Wolfram Media/Cambridge University Press, 1999.
27. W Wu. Basic principles of mechanical theorem proving in geometries. *Journal of System Sciences and Mathematical Sciences*, 4(3):207–235, 1984.
28. J Zhang. MCS: Model-based conjecture searching. In *Proceedings of CADE-16, LNAI, 1632*, pages 393–397. Springer Verlag, 1999.