Automated Theory Formation for Tutoring Tasks in Pure Mathematics

Simon Colton, Roy McCasland and Alan Bundy
Division of Informatics
University of Edinburgh, UK
simonco@dai.ed.ac.uk, rmccasla@dai.ed.ac.uk,
bundy@dai.ed.ac.uk

Toby Walsh
Cork Constraint Computation Centre
University College Cork, Ireland
tw@4c.ucc.ie

Abstract

The HR program forms mathematical theories from as little information as the axioms of a domain. The theories include concepts with examples and definitions, conjectures, theorems and proofs. Moreover, HR uses third party mathematical software including automated theorem provers and model generators. We suggest that a potential role for theory formation systems such as HR is as an aid to mathematics lecturers. We discuss an application of HR to the generation of a set of group theory exercises. This forms part of a project using HR to make discoveries in Zariski spaces, which is also detailed.

1 Introduction

At primary and secondary school levels, there are programs which generate exercise sheets automatically as an aid to mathematics teachers and students. In these cases, the emphasis is on asking the student to perform computations. At the university level, more emphasis is placed on using deduction in exercises. Hence, there is much potential for automated reasoning and other tools to be used in mathematics education, in particular for generating tutorial exercises.

We discuss here a project which couples a research mathematician with various pieces of mathematical software. In particular, the project aims to employ the HR automated theory formation system [3] to produce novel results in the domain of Zariski spaces [12]. This project, while still in its early stages, has produced both some promising preliminary results and an insight into the usage of HR by a research mathematician. In §3 we discuss this project and

the preliminary results from it, and in §4 we explain the improvements to HR which have been implemented in response to the user's suggestions. These improvements have enabled an application of HR not to a discovery task, but rather to a tutoring task: the aiding of a mathematician in producing a set of exercises in group theory. This application is discussed in §5.

2 The HR Program

The HR program (named after mathematicians Hardy and Ramanujan) performs automated theory formation in domains of pure mathematics such as number theory, graph theory, and finite algebras, such as group theory and ring theory. The initial information about a domain supplied to HR includes the axioms of the domain and optionally some initial concepts (e.g., multiplication and addition in number theory) and some objects of interest (e.g., some integers in number theory, groups in group theory, etc.). The concepts are supplied with both a definition and some examples (e.g., triples of integers related by multiplication). In finite algebraic domains, HR can start with just the axioms of the theory, as it extracts initial concepts from these, e.g., given the identity axiom in group theory, HR extracts the concept of identity elements.

HR operates by performing a theory formation step that attempts to invent a new concept from one (or two) old ones. Concept formation is facilitated by a set of general production rules that generate both a definition and set of examples for the new concept, from the definition and examples of the old concept(s) respectively. The *complexity* of a concept is measured as the number of production rule steps which were used to construct the concept. Similarly, the *applicability* of a concept is measured as the proportion of objects of interest in the theory for which the concept has a non-empty set of examples.

The production rules are described in detail in [3], [6] and [7]. In summary, these rules perform the following constructions:

- Compose rule: uses conjunction to join the definitions of two previous concepts
- Disjunct rule: uses disjunction to join the definitions of two previous concepts
- Equals rule: imposes equality in a concept's definition
- Exists rule: introduces existential quantification
- Forall rule: introduces universal quantification
- Match rule: equates variables in a concept's definition
- Negate rule: negates predicates in a concept's definition
- Size rule: counts subobjects
- Split rule: instantiates variables in a concept's definition

Figure 1 shows the construction of the concept of central elements in group theory.

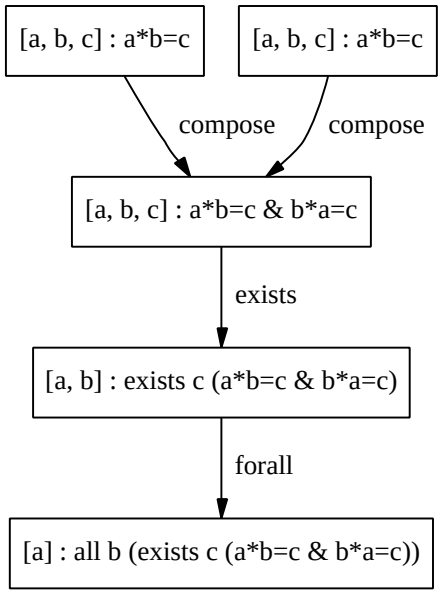

Figure 1: Construction of the concept of central elements in group theory

A theory formation step will lead to either: (a) a concept that has no examples, (b) a concept that has exactly the same examples as a previous concept, or (c) a concept that has non-trivial examples that differ from those of all previously existing concepts. In the first case, there may be no examples for the concept because of the lack of data given to HR, or it may be because the definition of the concept is inconsistent with the axioms of the domain. Hence, HR makes a conjecture that no examples of the concept exist. In the second case, HR makes an if-and-only-if conjecture, stating that the definitions of the new concept and the previous one are equivalent. In the last case, the concept is simply added to the theory.

When HR makes conjectures in finite algebraic domains, it can invoke the Otter theorem prover [13] to attempt to prove the conjecture. If Otter fails, HR invokes the MACE model generator [14] to attempt to find a counterexample. If neither Otter nor MACE are successful, then the conjecture remains open. In cases where Otter proves an equivalence theorem, HR breaks this into a set of implication theorems, where a set of premise predicates imply a single goal predicate. Furthermore, HR extracts prime implicates from each implication theorem, i.e., it takes ever-larger subsets of the premises from the implication theorem and sees whether Otter can prove that they imply the goal.

For instance, if it made the conjecture that

$$A \wedge B \wedge C \leftrightarrow D \wedge E \wedge F,$$

then HR would extract implicates such as these:

$$A \wedge B \wedge C \rightarrow D$$

$$D \wedge E \wedge F \rightarrow A$$

etc. From the first of these, HR would attempt to extract prime implicates in this order:

$$A \rightarrow D, \;\; B \rightarrow D, \;\; C \rightarrow D$$

$$A \wedge B \rightarrow D, \;\; A \wedge C \rightarrow D, \;\; B \wedge C \rightarrow D$$

$$A \wedge B \wedge C \rightarrow D$$

stopping when it found one that Otter could prove. For more details of how HR makes conjectures, see [4].

3 The Zariski Spaces Project

3.1 Zariski Spaces

Zariski Spaces were introduced in 1998 [12]. In order to understand these spaces, one needs to first understand Zariski Topologies. In a broad sense, these topologies are rather like prime factorizations. For example, the Zariski topology associated with the ring of integers consists of sets (called *varieties*) of prime ideals, one set for each integer. In particular, the variety of the integer 12 would be the set of ideals generated by 2 and by 3, respectively, since 2 and 3 are the prime factors of 12. Note that since 2 and 3 both divide 12, then the ideal generated by 2 and the ideal generated by 3 both contain the ideal generated by 12.

In the general case, let R be a commutative ring with unity, and let $specR$ denote the collection of prime ideals of R. Now for each (possibly empty) subset A of R, let the *variety* of A be given by: $V(A) = \{P \in specR : A \subseteq P\}$. It is easily shown that the collection of all such varieties constitutes (the closed sets of) a topology, called the Zariski Topology on R, which we denote by $\zeta(R)$. It turns out that every topology is a semiring, if one takes the operations of addition and multiplication to be set-theoretic intersection and union respectively. Now let M be an R-module, and repeat the above process. That is to say, let $specM$ denote the collection of all prime submodules of M, and for each subset B of M, let the variety of B be given by: $V(B) = \{P \in specM : B \subseteq P\}$. Finally, let $\zeta(M)$ represent the collection of all varieties of subsets of M. Then one can show that while $\zeta(M)$ seldom forms a topology, it does form a semimodule over the semiring $\zeta(R)$, where the operation in $\zeta(M)$ is taken to be intersection, and scalar multiplication is given by $V(A)V(B) = V(RAB)$.

There are reasons to suppose that Zariski Spaces might turn out to be of considerable importance in mathematics. For instance, Zariski Topologies and the study of varieties have played an important role in the development of Algebraic Geometry, and in particular, the Hilbert Nullstellensatz, which is one of the fundamental results in Algebraic Geometry. It is certainly possible that

Zariski Spaces could have a similar impact on some branch of mathematics. Furthermore, some preliminary results suggest a possible connection between a certain concept in Zariski Spaces, called subtractive bases, and direct sum decompositions within a large class of modules. The search for direct sum decompositions has been a major undertaking in mathematics for some time, and has so far proven quite intractable, except in special cases. The study of semimodules in general has already yielded many applications to computer science [10], and since Zariski Spaces are first and foremost semimodules, it is possible that their study will promote further advances in theoretical computer science.

3.2 Proposed Discovery Methods

At present, HR is mostly autonomous, i.e., the user sets some parameters for the search it will perform, then HR builds a theory and the user employs various tools to extract information about the theory. We propose to extend the theory of theory formation upon which HR is based by enabling any of the functionality of HR to be replaced on occasion by human intervention. That is, we intend to make HR semi-automated by allowing the user to provide proofs and counterexamples to conjectures HR makes and to specify related concepts and conjectures to base the theory formation around. Alongside the development of HR's functionality, we also intend to develop the application to Zariski spaces. Zariski spaces represent a higher level of complexity than the domains in which HR has so far been applied, and the new application will require an incremental approach whereby (i) HR is enabled to form theories about increasingly complicated domains related to Zariski spaces (ii) testing is performed to see if HR invents various concepts and conjectures required for it to proceed and (iii) theory formation is centred around the important concepts and the results analysed for any discoveries. The proposed route to Zariski spaces is via: semigroups, semirings and semimodules, followed by groups, rings and modules and finally, using a cross domain approach, Zariski spaces.

It is unclear at the moment how HR will interact with the user and with third party pieces of mathematical software on this project. It seems likely that HR will be used for an initial exploration of each domain, to: (a) invent core concepts (b) prove some fundamental theorems using a theorem prover (HR has access to Otter, E, Spass and Bliksem through the MathWeb Software Bus [9]), and (c) generate some examples of the concepts using a model generator or computer algebra system (HR uses MACE and Maple, also possible through MathWeb). Following the initial investigation, the user will both prune uninteresting concepts and specify which concepts should be emphasised in the next theory formation session. The user will be more involved in that session, choosing to prove theorems, provide counterexamples and direct the search where appropriate. We hope that improving HR to enable such an interaction will lead to the discovery of new results about Zariski spaces.

3.3 Preliminary Results

We have undertaken a pyramid approach to this project whereby we employ HR in successively more complex domains, eventually arriving at Zariski spaces. Along the way, we may find interesting results in the less complex domains. A particularly interesting result which was previously unknown to us occurred in the semigroup domain, which are algebras with a single associative operation. HR produced the conjecture, which Otter then proved, that:

$$c * c = c \ \& \ \exists \ d \ (c * d = b) \rightarrow c * b = b.$$

Paraphrased, this states that, given an idempotent element c, then if any element b appears in any column in the "c" row of the Cayley table of the given semigroup, then that same element b must also appear in the "c" row *in its own column*. Admittedly, this result appeared to us at first glance to be untrue, but upon reflection, the proof was actually fairly obvious. This result is of some interest to us, because Zariski Spaces are a rather complicated algebraic structure, but at their simplest level, they are a special kind of semigroup. Moreover, every element in these spaces is an idempotent element, and therefore, this theorem is applicable to every element.

4 Additions to HR Arising from the Project

Each new application of HR necessitates some additional functionality, and the application to Zariski spaces is no exception. Firstly, we have implemented a new production rule so that HR can check whether particular concepts exhibit certain algebraic properties. Secondly, we have introduced a reaction mechanism so that we can tailor HR's search to react to a particular event, such as the introduction of a concept with a particular property. Thirdly, we have enabled HR to use Knuth-Bendix completion so that it can identify (and discard) a range of conjectures which are uninteresting. These additions are described in more detail below.

4.1 The Embed-Algebra Production Rule

All of HR's conjectures discuss concepts which it has invented, and HR uses only a small number of production rules to generate new concepts. Hence the implementation of a new production rule represents a major addition to HR's functionality. Each production rule must:

- Determine the ways in which it can be used to produce new concepts from a given set of old concepts, in terms of the parameterisations of the rule possible for the old concept(s).

- Produce a complete set of examples for the new concept for each object of interest (group, graph, integer, etc.) in the theory.

- Produce a definition for the concept which is consistent with the examples.

Drawing on work by Graham Steel [16] [17] into cross domain theory formation, we implemented the 'embed- algebra' production rule, which aims to find algebraic structures embedded in old concepts. Any concept which is a predicate of arity 4, where the first entry is an object of interest and the following three entries are of the same type, can be tested for various algebraic structures. For instance, if HR invented a concept with a predicate definition $P(a, b, c, d)$ such that b, c and d were subobjects of a of the same type, then the embed-algebra rule could be invoked to check whether the function $f(b, c) = d$ had particular algebraic properties, such as commutativity, associativity, an identity, etc.

This production rule differs from all the others in that it relies on a third party program, MACE, to generate the examples of the new concept. Given an old concept C of arity 4, MACE is used to check whether the set of triples for each object of interest that C applies to satisfies a set of axioms chosen by the user. For example, suppose that concept C had the following datatable:

object of interest	subobject 1	subobject 2	subobject 3
o_1	s	s	s
o_2	t	t	t
o_2	t	u	u
o_2	u	t	u
o_2	u	u	t
o_3	a	a	b
o_3	a	b	b
o_3	b	a	b
o_3	b	b	b
o_4	x	z	z
o_4	x	y	x
o_4	y	x	y

In this case, the embed-algebra rule takes each object of interest o in turn and collects all ground instances of C as examples of the operation upon which the axioms are to act. For instance, HR takes object o_2 and collects the following four instances of the operation upon which the axioms are going to be checked:

$$t * t = t, \;\; t * u = u, \;\; u * t = u, \;\; u * u = t$$

HR first normalises the elements so that they are represented by the numbers 0, 1, 2, etc. It then checks that it has not seen this (possible) algebra before, and if it has, uses the results of the previous calculation.

If the set is new, HR then checks whether or not this operation is closed, by checking that every element which appears as the product of an operation also appears as both the left hand and right hand element in a product and that the examples constitute a Cayley table. If the operation passes this check, then MACE is invoked to check whether the operation satisfies the axioms chosen by the user. To use MACE in this manner, both the axioms and all the ground instances of the operation are passed to MACE. Hence, for o_2 above, if the user chose the group theory axioms, MACE would be passed the following file:

```
set(auto).
formula_list(usable).
0*0=0.
0*1=1.
1*0=1.
1*1=0.
all a (a*id = a & id*a = a).            \\ identity
all a (a*inv(a) = id & inv(a)*a = id).  \\ inverse
all a b c ((a*b)*c = a*(b*c)).          \\ associativity
end_of_list.
```

If MACE responds that it has found a model, then this effectively states that the model HR has supplied satisfies the axioms chosen. Hence, the operation (and thus the original concept) embeds an algebra for the particular object of interest under consideration.

In the current example, if the user had chosen the axioms for group theory, then the operation for the elements of o_1 and o_2 would both be closed and satisfy the axioms, but the operation for o_4 is not closed (z does not appear in the product anywhere) and the operation for o_3 does not have an identity element, which is required for the group theory axioms. To construct the datatable for the new concept, for each object of interest for which the operation does embed an algebra, HR first checks whether any algebra it has already is an isomorphism of the embedded algebra, and substitutes the previous one if so. Then the datatable is constructed with the objects of interest in the first column and the algebra embedded in the second column. If the embedded operation does not satisfy the axioms of the algebra, then an empty set is put in the second column.

If the concept which was input to this production rule had concept number F, and the algebra chosen to check for embedding was Alg, then the definition for the new concept would be generated as:

$$[a, b] : \text{concept } F \text{ forms } Alg\ b \text{ for } a$$

for instance:

$$[a, b] : \text{concept } 17 \text{ forms group } b \text{ for } a$$

Note that the user can specify that HR checks for various algebras using this rule. These choices comprise the different parameterisations of the embed-algebra production rule that HR will carry out.

4.2 Reaction Mechanism

A motivating example for the embed-algebra production rule was for HR to find the centre of groups, not just as a subset of elements, but also as a subgroup. However, as we see from figure 1 above, HR invents the concept of a central element as an element type, not as a concept of arity four in which an operation could be embedded. It is not controversial to state that, if a mathematician invents a new type of element in an algebra like group theory, one of the first

things they might try is to see whether the set of elements forms a subgroup for all groups. We wanted to model this kind of reaction to a new element type in HR, and – to cover the more general case – we implemented a 'reactive' search in HR. Such a search is similar to the way in which certain "demons" are invoked when particular slots are filled in frame representations of concepts [2].

The reactive search in HR is determined beforehand by the user, who supplies pieces of pseudo-java code to HR that specify what HR should do if certain things occur during its normal theory formation. Using Java's reflection mechanism, HR translates these pieces of code (which we call 'reactions') into Java code at runtime, and the conditions for them are checked whenever a new theory formation step is completed; a new concept is introduced; a new conjecture is made or a new counterexample is found.

For instance, the following reaction reacts to a new concept being introduced:

```
react_to(concept)
condition(concept, applicability > 0.8)
condition(concept, arity == 3)
action(add_to_agenda, n1 = concept exists [2])
action(add_to_agenda, n2 = n1 size [1])
action(mark_concept, n2, interesting)
```

With this reaction activated, for each new concept, HR first checks whether the applicability[1] of the new concept is greater than 0.8. If so, HR then adds two steps to the agenda, the first of which performs a step using the exists production rule on the new concept, and the second of which performs a step using the size production rule on the concept produced by the first step (n1). Note that if any step results in a previous concept, this is substituted in further steps added to the agenda. Finally, this reaction tells HR to mark the concept produced by the second step (n2) as interesting, so that the user can find all such concepts in HR's output.

HR's reaction mechanism is still under development. As discussed in §5 below, we use the reaction mechanism to force HR to check immediately whether a new type of element forms a subgroup in a group theory session. This is a generalisation of work done with Alison Pease [15] on getting HR to react in ways prescribed by Lakatos in [11].

4.3 Knuth-Bendix Completion to Discard Conjectures

A problem we have come across throughout the development of HR is that it produces too many conjectures of a trivial nature, as lamented in chapter 11 of [3]. We have previously relied on Otter's proof length statistic to prune conjectures which Otter found easy to prove, but we have recently found this approach can discard fairly interesting conjectures, and so is not appropriate.

Instead, to reduce the number of trivial results produced by HR, one can use a set of rewrite rules as a filter. Otter can, in some cases, be used to

[1] See [8] for a description of HR's measures of interestingness, including the applicability of a concept.

produce a set of rewrite rules (demodulators) for certain collections of axioms. In particular, we can use the Knuth-Bendix option in Otter to derive a list of equations from the axioms of an algebra, and these equations form a complete set of rewrite rules for the given axioms. To be more specific, in group theory, we give Otter the group axioms of left identity, left inverse and associativity, and Otter outputs the following list of 10 equations:

```
id*x=x.            inv(x)*x=id.     (x*y)*z=x*y*z.
inv(x)*x*y=y.      x*id=x.          inv(id)=id.
inv(inv(x))=x.     x*inv(x)*y=y.    x*inv(x)=id.
inv(x*y)=inv(y)*inv(x).
```

On inspection, this list includes the right-hand versions of identity and inverse, as well as these facts: the identity is its own inverse, the inverse composed with itself is the identity function, and the inverse of a product is the product of the inverses in reverse order. Having obtained such a list of equations, HR take each prime implicate it produces and sends the premises of that conjecture to Otter, which is told to work in Knuth-Bendix mode. Otter then uses its demodulator and paramodulator functions to simplify the premises. Sometimes, this rewriting process includes in its output the goal of the original conjecture. In these cases, HR discards the conjecture as it is considered trivial.

As an example, HR produced the following conjecture in a group theory session: $b * c = d \& b = id \rightarrow b * d = c$. This is a true statement in group theory. When we gave the premises of this conjecture to Otter in Knuth-Bendix mode, it produced several conclusions, including the one conjectured here. It should be noted that a human would typically arrive at the conclusion by first observing that since $b = id$, the first premise yields $c = d$. At this point, the conclusion would be obvious. While one can perhaps argue that this thought process is not an entirely straightforward one, and that therefore the conjecture has some merit, the result is not sufficiently important to be included in the theory, and therefore a human mathematician would probably discard it. Thus, HR's decision to discard the conjecture matches that of human mathematicians.

To test this new method, we ran HR for 1000 theory formation steps and it produced 281 prime implicates, of which 40 (14%) were discarded because Knuth Bendix completion found the goal from the premises. For example, HR discarded the conjecture $b*c = d \& c*b = d \& d*b = c \rightarrow b*d = c$, because, using the demodulators extracted from the axioms of group theory, Otter's Knuth-Bendix mode managed to re-write the premises to include the goal. We are currently working to get Otter to discard more conjectures of a similarly trivial nature. This is similar to our efforts described in [5], where we ask HR to discard conjectures about Maple [1] functions if Otter is able to prove them, as these will most likely be trivially true.

5 Application to Setting Exercises

The second author of this paper has much experience in mathematics education at the university level, and it was our aim to see if HR alongside Otter and MACE could aid him in the setting of tutorial questions for an undergraduate group theory course.

Based on the observation that many such tutorial questions ask the student to show that a particular set of elements form a subgroup, we decided that HR's embed algebra production rule should be used to identify subgroup types. To facilitate this, we wrote a reaction script which ensured that, whenever HR invented a new type of element, it formed the subgroup (where possible) of those elements, and flagged those to the user for which: (a) the subset always formed a subgroup and (b) the subgroup generated was neither the trivial group with one element nor the entire parent group for at least one of the groups in HR's database. The following reaction script achieved this:

```
react_to(concept)
condition(concept,types.toString() == [group, element])
action(add_to_agenda,n1 = group003 concept compose [1,2,0,0])
action(add_to_agenda,n2 = n1 concept compose [1,0,2,0])
action(add_to_agenda,n3 = n2 embed_algebra [group])
condition(n3,applicability == 1)
action(add_to_agenda,n4 = n3 match [0,0])
action(add_to_agenda,n5 = n3 split [[1],[0]])
condition(n4,applicability < 1)
condition(n5,applicability < 1)
action(mark_concept,n3,good_subgroup)
action(mark_concept,concept,good_element_type)
```

We gave HR the groups up to order 8 to use as data for the empirical conjectures that it made, and formed a theory using 10,000 theory formation steps. We used the compose, exists, match and equals production rules in a breadth first manner, and controlled the use of the forall production rule via another reaction script. This forced HR to use the forall rule only with concepts x which relate two elements, to invent the concept of elements for which all the other elements are related to x. We have found that normal usage of the forall production rule often leads to many complicated – but generally uninteresting – concepts in group theory, so we controlled it with a reaction script instead.

The user was allowed to write any tutorial questions, as long as they were inspired in some way by something from HR's theory. On reflection of how HR's subgroup concepts were being used to generate tutorial questions, we identified the following possibilities:

• For those element types, C, which the user can prove (possibly using Otter) always form a subgroup, the teacher may set the exercise: "Prove that elements with property C form a subgroup".

• For those element types, C, for which HR has found groups where they don't form a subgroup, and, indeed, for those where such a group is later found by MACE, the teacher could ask the following question: "Characterise those groups for which the set of elements with property C form a subgroup." Note, however, that characterisation problems can often be very difficult, and sometimes of little interest to mathematics.

• For those element types where the smallest group for which the elements do not form a subgroup is fairly large (i.e., beyond the reach of simple computation), the teacher could ask the following: "Find the smallest group for which the set of elements with property C do not form a subgroup". As the answer is large, the student will be forced to use their knowledge of group theory to answer the question, because a calculation would not easily yield the answer. Note that questions of this type may also be difficult to answer.

In addition, the mathematician may choose to simplify the questions by restricting the range of groups to which a question applies. For example, they may ask: "Show that, for Abelian/cyclic/dihedral groups, the set of elements with property C form a subgroup". Finally, other questions may arise as part of this process, and HR may be able to inspire such questions, with MACE and Otter possibly providing answers to such questions.

5.1 Results

We ran HR on a 2.0 Ghz. Pentium 4, for 10000 theory formation steps, giving as input the datatables for all groups of order 8 or less. The run took 301 seconds and produced a total of 330 concepts. Of these concepts, 17 were element types which produced subgroups for at least some of the initial groups. Of these 17, 10 concepts produced subgroups for all of the initial groups. Of these last 10, 8 concepts produced at least one non-trivial subgroup; that is, a subgroup which was neither the identity subgroup, nor the group itself. The concept definitions of this last type were as follows:

```
g43. [a, b] : exists c (c*c=b)
g93. [a, b] : all c (exists d (d*c=b & c*d=b))
g102. [a, b] : exists c d (b*c=d & d*b=c)
g110. [a, b] : all c (b*(c*b)=c)
g124. [a, b] : exists c (c*(c*b)=b & inv(c*b)=(c*b))
g224. [a, b] : inv(b)=b & (exists c (c*c=b))
g282. [a, b] : exists c (inv(c)=b & c*c=b)
g307. [a, b] : exists c ((c*c)*(c*c)=b)
```

The remaining 9 concepts produced subgroups for some, but not all of the initial groups.

```
g10. [a, b] : all c (inv(c)=b)
g16. [a, b] : all c (c*c=b))
```

*	0	1	2	3	4	5	6	7	8	9	10	11
0	0	1	2	3	4	5	6	7	8	9	10	11
1	1	0	3	2	5	4	7	6	9	8	11	10
2	2	3	4	5	0	1	8	9	10	11	6	7
3	3	2	5	4	1	0	9	8	11	10	7	6
4	4	5	0	1	2	3	10	11	6	7	8	9
5	5	4	1	0	3	2	11	10	7	6	9	8
6	6	7	10	11	8	9	1	0	5	4	3	2
7	7	6	11	10	9	8	0	1	4	5	2	3
8	8	9	6	7	10	11	3	2	1	0	5	4
9	9	8	7	6	11	10	2	3	0	1	4	5
10	10	11	8	9	6	7	5	4	3	2	1	0
11	11	10	9	8	7	6	4	5	2	3	0	1

Figure 2: Group produced by MACE disproving subgroup closure property

```
g29. [a, b] : inv(b)=b
g67. [a, b] : exists c d (c*d=b & (all e (e*e=c)))
g198. [a, b] : all c ((exists d (d*c=b)) & (exists e (e*e=c)))
g202. [a, b] : all c (c*b=c & (exists d (d*d=c)))
g262. [a, b] : all c (exists d (d*c=b & d*d=c))
g267. [a, b] : exists c (c*(c*c)=b)
g303. [a, b] : all c ((c*c)*(c*c)=b)
```

In the course of setting the exercises, we looked at the first two concepts (g43 and g93) in more detail. Concept g43 simply defines the set of all elements which appear on the diagonal of the Cayley Table of the group. This concept gives a subgroup for each of the groups of order 8 or less, as demonstrated by HR using MACE during the theory formation. However, we were fairly certain that this construction does not always yield a subgroup. We therefore asked MACE to generate a counterexample, by asking it to find a group for which there are two elements on the diagonal which multiply to give an element which is not on the diagonal. If it could find one, then the subset would not be closed under multiplication and hence not a subgroup. We were unsuccessful in finding a counterexample up to order 11, but at order 12, MACE took 142.84 seconds of cpu time to generate the non-Abelian group in figure 2. We see that elements $0, 1, 2$ and 4 appear on the diagonal, but the product of 1 and 2 is 3 which is not on the diagonal. Hence concept g43 does not form a subgroup for this group.

It is perhaps not entirely obvious that the second concept in the top list, g93, actually generates the centre of a group. To see this, first solve both equations for d, to give $d = b * inv(c)$ and $d = inv(c) * b$, or more succinctly, $b * inv(c) = inv(c) * b$. Now note that since the concept applies to all elements c in the group a, we can substitute $inv(c)$ for c, and it becomes clear that the concept is equivalent to: $[a, b] : \forall\ c\ (b * c = c * b)$. We asked Otter to prove that this set is a subgroup, by giving it the concept and the group axioms, and then

Axiom	Statistic	Value
Identity	Proof Length	1
	Proof level	1
	Cpu time	0.20 seconds
Inverse	Proof Length	36
	Proof level	10
	Cpu time	24.67 seconds
Closure	Proof Length	59
	Proof level	14
	Cpu time	54.55 seconds

Table 1: Otter statistics from proving that central elements form a subgroup

having it prove, each in turn, that the concept definition is closed with respect to the products, the inverses and the identity. That is, we ran Otter three times, first with the group axioms and these two additional lines:

```
all a (f(a) <->  (all b (((exists c (c*b=a & b*c=a))))).
-(all a b (f(a) & f(b) -> f(a*b))).
```

[Note that the goal is negated as Otter is a resolution prover]. This proved that the subset is closed under multiplication. Secondly, we ran Otter with the group axioms and these two additional lines:

```
all a (f(a) <->  (all b (((exists c (c*b=a & b*c=a))))).
-(all a (f(a) -> f(inv(a)))).
```

This proved that the inverse axiom applies to the subset. Finally, we ran Otter with the group axioms and these two additional lines:

```
all a (f(a) <->  (all b (((exists c (c*b=a & b*c=a))))).
-(exists a (f(a) & (all b (f(b) -> (a * b = b & b * a = b))))).
```

This proved that there is an identity element in the subset. Given that the parent multiplication operation was inherited by the subset, we had no need to prove associativity in the subgroup. Hence, with the three proofs completed, we had shown that the centre of a group forms a subgroup. The statistics Otter used in this way are given in table 1.

Having run HR and explored the results as described above, the mathematician then set the exercises, with the resulting exercise sheet given in appendix A. It is difficult to assess the success of this work in objective terms, and we have included in appendix A a brief summary of how the exercises were influenced by HR's theory. We hope that this demonstrate that HR, Otter and MACE contributed in a non-trivial way to the setting of the exercises, and that there is much potential for similar use in future.

6 Conclusions and Future Work

We have demonstrated that the HR theory formation program, in conjunction with the Otter theorem prover and MACE model generator can be used to help write tutorial questions for mathematics students. The programs acted in the role of an aid to a mathematics teacher, rather than as an autonomous mathematician. In particular, HR was used to empirically suggest subgroup constructions, and HR called MACE as part of this process. Furthermore, Otter and MACE were both used to check whether certain element types form a subgroup in the general case. We documented a case where Otter proved that a subset of elements always forms a group and a case where MACE showed that another subset of elements doesn't necessarily form a group (as HR had suggested based on the evidence of the groups up to order 8).

To undertake this application, we implemented two new functionalities, namely a reactive search, where HR's search can be interrupted in response to certain events, and a new production rule which identifies when certain algebras are embedded in old concepts. We believe that a sustained application of HR in this manner to other algebras could yield further interesting tutorial questions. However, the main aim of this project is to use HR to discover new results in the domain of Zariski spaces, and we are already getting positive results and feedback from this project. To succeed further in this project, our future work will centre around:

• The implementation of more user interface functionality, so that the user can act themselves as the concept formation/conjecture making/theorem proving/counterexample finding process.

• Storage of knowledge between sessions. This will be important in order to (i) reduce the time spent by the mathematician sifting through results he/she has already come across and (ii) import and apply in a new context important results derived during previous sessions.

• Further exploration of Zariski spaces. This will proceed as planned – HR will be used in successively more complicated domains until eventually able to work with Zariski spaces.

• Further pruning of uninteresting conjectures. This will be achieved by using Knuth-Bendix completion more fully and via a variety of other techniques, some using automated theorem provers such as Otter.

We believe that for automated deduction, the role of an aid for mathematics lecturers is a more plausible prospect in the short term than the role of a discoverer of important new proofs. However, such an application needs to be fuelled by automated theory formation in order to identify interesting results that the lecturer would possibly miss otherwise.

Acknowledgments

This work is supported by EPSRC grants GR/M98012 and GR/R84559/01. We would like to thank Derek Sleeman for his interesting comments about this work. We would also like to thank Alison Pease for her input to the reaction mechanism and other aspects of HR's development. Simon Colton is also affiliated with the Department of Computer Science at the University of York.

References

[1] M. Abell and J. Braselton. *Maple V by Example.* Associated Press Professional, 1994.

[2] A. Barr, P. Cohen, and E. Feigenbaum, editors. *The Handbook of Artificial Intelligence, IV.* Addison-Wesley, 1989.

[3] S. Colton. *Automated Theory Formation in Pure Mathematics.* PhD thesis, Department of Artificial Intelligence, University of Edinburgh, 2000.

[4] S. Colton. The HR program for theorem generation. In *Proceedings of the Eighteenth Conference on Automated Deduction*, 2002.

[5] S. Colton. Making conjectures about maple functions. In *Proceedings of the Tenth Symposium on the Integration of Symbolic Computation and Mechanized Reasoning*, 2002.

[6] S. Colton, A. Bundy, and T. Walsh. HR: Automatic concept formation in pure mathematics. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999.

[7] S. Colton, A. Bundy, and T. Walsh. Automatic identification of mathematical concepts. In *Machine Learning: Proceedings of the Seventeenth International Conference*, 2000.

[8] 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.

[9] A. Franke and M. Kohlhase. System description: MATHWEB, an agent-based communication layer for distributed automated theorem proving. In *Proceedings of Sixteenth Conference on Automated Deduction*, pages 217–221, 1999.

[10] J. Golan. *The theory of Semirings with Applications in Mathematics and Theoretical Computer Science.* John Wiley and Sons, 1992.

[11] I. Lakatos. *Proofs and Refutations: The logic of mathematical discovery.* Cambridge University Press, 1976.

[12] R. McCasland, M. Moore, and P. Smith. An introduction to Zariski spaces over Zariski topologies. *Rocky Mountain Journal of Mathematics*, 28:1357–1369, 1998.

[13] W. McCune. Otter 3.0 Reference Manual and Guide. Technical Report ANL-94/6, Argonne National Laboratory, Argonne, USA, 1994.

[14] W. McCune. MACE 2.0 Reference Manual and Guide. Technical Report ANL/MCS-TM-249, Argonne National Laboratory, Argonne, USA, 2001.

[15] A. Pease, S. Colton, A. Smaill, and J. Lee. Lakatos and machine creativity. In *Proceedings of the ECAI Creative Systems Workshop*, 2002.

[16] G. Steel. Cross domain concept formation using HR. Master's thesis, Division of Informatics, University of Edinburgh, 1999.

[17] G. Steel, S. Colton, A. Bundy, and T. Walsh. Cross domain mathematical concept formation. In *Proceedings of the AISB-00 Symposium on Creative & Cultural Aspects and Applications of AI & Cognitive Science*, 2000.

A Group Theory Exercise Sheet

The following exercise sheet was producing using the theory HR formed as described in §5.1 above. Below, we give a brief summary of how HR's results inspired these questions.

Let G be a multiplicative group with identity 1. Problems marked ** should be considered honors exercises.

1. Let $A = \{a \in G :$ for some $c, d \in G$, $ac = d$, $ad = c$ and $c^{-1} = c\}$ and let $B = \{b \in G : b^2 = 1\}$. Prove or disprove that $A = B$. Determine whether A is a subgroup of G. If so, prove it. If not, give the smallest example of G such that A is not a subgroup of G.

** 2. Let $A = \{a \in G : a = b^2$ for some $b \in G\}$. Determine whether A is a subgroup of G. If A is not a subgroup, give a characterisation for the groups G for which A is a subgroup.

3. Let $A = \{a \in G : ab = ba$ for all $b \in G\}$. Prove or disprove that A is a subgroup of G.

4. Let $A = \{a \in G :$ for some $c, d \in G$, $ac = d$ and $da = c\}$ and let $B = \{b \in G :$ $bcb = c$ for some $c \in G\}$. Prove or disprove that $A = B$. Determine whether A is a subgroup of G. If so, prove it. If not, give the smallest example of G such that A is not a subgroup of G.

5. Let $A = \{a \in G : aca = c$ for all $c \in G\}$. Recall that if $ac = c$ for some $c \in G$, then $ac = c$ for all $c \in G$. With this in mind: Determine whether the set A is equal to B in exercise 4. Determine whether A is a subgroup of G.

6. Let $A = \{a \in G$: for some $x \in G$, $x^2a = a$ and $(xa)^{-1} = xa\}$. Let $B = \{b \in G : b = b^{-1}\}$. Let $C = \{c \in G : cyc = y$ for some $y \in G$ such that $y^2 = 1\}$. Is $B \subseteq A$? Is $A \subseteq B$? Is $A = C$? Is this A the same as the A in exercise 4? Prove your answers. Determine whether A is a subgroup of G.

7. Let $A = \{a \in G$: for some $c, d, f \in G$, $ac = d$, $ad = c$, $a = f^2$ and $c = c^{-1}\}$ and let $B = \{b \in G : b^2 = 1$ and $b = g^2$ for some $g \in G\}$. Is $A = B$? How does this A compare with the A in exercise 1? Is this A a subgroup of G? Prove your answers.

8. Let $A = \{a \in G : a = b^3$ for some $b \in G\}$. Determine whether A is a subgroup of G. If so, prove it. If not, give the smallest example of G such that A is not a subgroup of G.

** 9. Characterise the groups G for which every element of G can be written as a fourth power, (i.e., for all $a \in G$, there exists $b \in G$ such that $b^4 = a$).

10. Let $A = \{a \in G : a = (a^{-1})^2\}$ and let $B = \{b \in G : b^3 = 1\}$. Is $A = B$? Is A a subgroup of G? Prove your answers.

A.1 Summary of Motivations

Referring to the concept numbers as specified in §5.1, in Question 1, set B is in fact equivalent to the set defined by concept g29, since $inv(b) = b$ iff $b * b = id$. Set A came from concept g30. These two concepts are indeed equivalent, and the proof thereof makes a reasonable exercise for undergraduate students. In Question 2, set A is clearly the set defined by concept g43, which is discussed in §5.1. Set A in Question 3 comes from concept g93, also discussed in §5.1.

Set A of Question 4 comes from concept g102, and the A of Question 5 is derived from g110. The set B of Question 4 was conceived by us, merely by simplifying the definition of the corresponding set A. Having looked at these three sets, we came up with the first part of Question 5. We feel that students need to realise that, even though the existential and universal quantifiers are clearly different, nevertheless, on occasion they turn out to produce the same results. It should perhaps be pointed out that on this occasion, they of course do make a difference. It seemed to us a good opportunity to force the students to think a little, rather than merely give an instinctive reaction.

In Question 6, the set A is given by concept g124, and set B is just concept g29 again. We wanted a situation where students would find that, although the two sets were not equal, one set was contained in the other. Set C resulted from combining this latter concept with the set B of Question 4. In Question 7, set B is derived from concept g224, and set A is simply a combination of the set A of Questions 1 and 2. The sets in Questions 8 and 9 are given by concepts g267 and g303, respectively. The set A in Question 10 is from concept g282, and set B is clearly just a rewrite of the definition for set A.