Agent Based Cooperative Theory Formation in Pure Mathematics

Simon Colton, Alan Bundy and Toby Walsh*

University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, United Kingdom.
*University of York, Heslington, York Y010 5DD, United Kingdom.

simonco@dai.ed.ac.uk, bundy@dai.ed.ac.uk, toby.walsh@cs.york.ac.uk

Abstract

The HR program, Colton et al. (1999), performs theory formation in domains of pure mathematics. Given only minimal information about a domain, it invents concepts, make conjectures, proves theorems and finds counterexamples to false conjectures. We present here a multi-agent version of HR which may provide a model for how individual mathematicians perform separate investigations but communicate their results to the mathematical community, learning from others as they do. We detail the exhaustive categorisation problem to which we have applied a multi-agent approach.

1 Introduction

Automated theory formation in pure mathematics involves the production of mathematical concepts, examples, conjectures, theorems and proofs. Various systems have modelled different aspects of theory formation. The AM program, Davis and Lenat (1982), worked in elementary number theory and modelled how an exploratory approach can drive theory formation. It used heuristics to guide the search towards more interesting concepts and achieved some success re-inventing well known concepts. The GT program, Epstein (1988) worked in graph theory and was the first to model the use of theorem proving to help direct theory formation. The IL program, Sims and Bresina (1989), constructed operators with given properties over types of numbers such as complex numbers. This modelled how theory formation can be goal directed.

The AM program stopped being productive after a while in every session. AM's author, Lenat, argued that this was because it needed more heuristics and the ability to invent its own heuristics. He implemented the Eurisko program to do this, Lenat (1983), but it but wasn't as successful as AM and it is debatable whether it added to our understanding of theory formation. In Furse (1990) several other reasons are given why AM 'ran out of steam'. One reason based on arguments from Kuhn (1970) is that AM does not model any social aspect of the mathematical community. That is, creativity in mathematical research often arises from the interaction of several mathematicians, often collaborating on the same problem but sometimes working on different problems, possibly even in different domains.

To our knowledge, no theory formation program in mathematics has modelled the communication of ideas between mathematicians. We have extended the HR program, Colton et al. (1999), to model limited interaction between different copies of the program running concurrently. This multi-agent approach has led to greater creativity in the system as a whole. In §2 we give necessary background information about the HR system, followed in §3 by details of the multi-agent implementation. In §4 we discuss the exhaustive categorisation problem to which we have successfully applied multi-agent theory formation. In §5 we discuss further possibilities for this approach, including an application to the problem of integer sequence extrapolation.

2 The HR System

The HR system models the major activities of mathematical theory formation: forming concepts, calculating examples, making conjectures, proving theorems and finding counterexamples. The version of HR discussed in Colton et al. (1999) and Steel et al. (2000) is a Prolog implementation which includes all of this functionality. Java is a more natural language for implementing agent based programs and the version of HR we discuss here is a re-implementation of HR in Java which is still under development. The Java version does not yet have conjecture making or theorem proving abilities, so theory formation is limited here to the compilation of concepts. HR does this by exploring a space of concepts using a best first search based on measures of interestingness.

The user supplies a set of **objects of interest** for the domain, eg. the numbers 1 to 10 in number theory. They also supply a set of initial concepts by providing a **definition** in terms of a set of predicates, the conjunction of which defines the concept, and an exhaustive **datatable** of examples calculated for all the objects of interest. For instance, the concept of multiplication in figure 1 is supplied in number theory with a datatable of examples where the first column contains integers, which the integers in the second and third columns multiply to give. A definition is also supplied for multiplication as a set of six pre-

dicates describing the triples $[n, a, b]$ in the datatable:

(i) n is an integer (ii) a is an integer (iii) b is an integer
(iv) a divides n (v) b divides n (vi) $a \times b = n$

Theory formation proceeds in **theory formation steps**: HR takes a concept already in the theory and passes it through a **production rule** (detailed below) along with a parameterisation detailing exactly what the production rule should do. The production rule will generate the definition and datatable of a new concept. HR then checks whether it has a concept in the theory already with exactly the same datatable. If it finds a match, the definition for the new concept is added as an alternative definition to the old concept, and the new concept is discarded. In fact, if the new concept is less complex (as defined below) than the original, HR replaces the original concept with the simpler new one. If the new concept does not match one already in the theory, it is added to the theory.

Currently HR has uses just 7 production rules. The types of concepts they produce is described briefly below. It is perhaps easiest to imagine that the objects discussed are integers and the subobjects are divisors.

- The **exists** rule produces concepts identifying objects where there exists a subobject of a particular nature.
- The **forall** rule produces concepts identifying objects where all subobjects are of a particular nature.
- The **size** rule produces functions which count the number of subobjects of a particular nature for each object.
- The **split** rule produces concepts identifying objects with a particular number of subobjects.
- The **match** rule produces concepts identifying objects with equal subobjects of a particular nature.
- The **negate** rule produces concepts identifying objects which have the properties described by one old concept but not the properties of another old concept.
- The **compose** rule produces concepts identifying objects which have the properties of two old concepts.

Note that the first five rules are called **unary** production rules as they produce a new concept from only one previous concept. The last two are called **binary** production rules as they produce a new concept from two previous ones. In figure 1 we see that to construct the concept of square numbers from the concept of multiplication, two theory formation steps are required. Firstly, the match production rule is used to construct the concept of integers and their *integer* square roots. Secondly, the exists production rule is used to identify those integers for which there exists such an integer square root, namely 1 and 4. For a more detailed description of the production rules, please see Colton et al. (2000a).

The **construction history** of a concept is the set of triples of (old concept, production rule, parameterisation) which detail the steps used to build all the previous concepts upon which the concept is based. Given the construction history of a concept and the user supplied concepts upon which it is ultimately based, HR can completely re-construct the concept by following the steps in order. We say the **complexity** of a concept is the size of its construction history. Even with just seven production rules, the size of the space HR searches when forming a theory is very large. To limit the search, we usually employ a complexity limit of between 5 and 10, ie. no theory formation steps involving concepts with complexity greater than the limit are allowed.

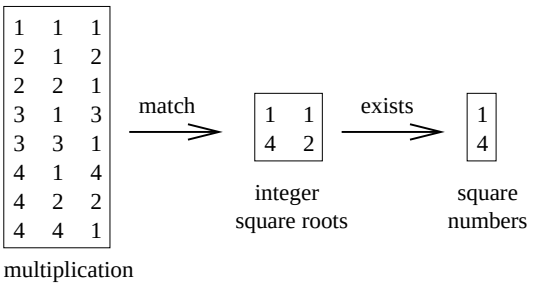

Figure 1: The construction of square numbers

Each new concept formed is added to the **agenda**. If a concept reaches the top of the agenda, all theory formation steps involving it are carried out until it is replaced at the top. HR can perform a **breadth first search** which puts every new concept to the bottom of the agenda, and a **depth first search**, which puts them at the top. Alternatively, HR can perform a **unary first** search which combines the breadth first and depth first searches. In a unary first search, HR uses the unary production rules in a depth first manner, but the binary production rules in a breadth first manner. This means that any new concept introduced is explored thoroughly with the unary rules, but not combined with other concepts until much later.

To enable effective traversal of the space we have enabled HR to employ a **best first search**: after every step it determines which is the most interesting concept and moves this to the top of the agenda. HR has many different measures available to estimate the interestingness of a concept, as detailed in Colton et al. (2000b), and uses a weighted sum of values calculated for a particular concept to determine the overall worth of the concept.

We discuss only the **novelty** measure here. To define this, we note that we can use the examples of a concept to produce a **categorisation** of the objects of interest in the theory. For instance, the square number concept in figure 1 categorises integers 1 to 4 as: $[1, 4], [2, 3]$ because 1 and 4 are squares, 2 and 3 are not. For every concept in a theory, HR can determine the categorisation it produces for the examples supplied by the user. The categorisation for a particular concept may not be unique and we define the novelty of a concept to be the reciprocal of the number of other concepts which share its categorisation. Therefore, concepts producing categorisations unique to them score 1 as they are novel, but concepts producing categorisations which also belong to 99 other concepts score $1/100 = 0.01$ as they are not novel.

3 Multi-Agent HR

Given a problem to solve, one approach is to employ a set of autonomous programs, called **agents**, each with different abilities and tasks and each able to communicate with the others. The set of agents forms an **agency** and it is hoped that dividing tasks between the agents will improve the overall efficiency of the system. Often each agent runs on a separate processor, and improvements in efficiency are observed as a result of the parallel attack on the problem. For our purposes, using an agency allows us to model a community of mathematicians each performing individual investigations but communicating their results to others.

Our implementation of an agency is fairly straightforward. Using the multi-threading capabilities in Java, we run several copies of HR as individual threads. Therefore our agency runs on the same processor, not on parallel processors, although this could easily be altered to improve efficiency. Each agent has a different name to identify the concepts they introduce, and each has different settings which guide its theory formation. We also run a 'watcher' program in another thread which determines when the task set for the agency has been achieved, and stops the agents when this is the case.

Communication between agents is limited to sending and receiving concepts. Each agent has a set of **inboxes** into which they receive concepts from the other agents, with a different inbox for each other agent. There is no global repository to which concepts are sent and taken, and the user can customise each agent to control which concepts from which inboxes it takes. It is hoped that the communication of a concept will increase the number of ways it is developed. For example, the binary rules combine two concepts. As the concepts available to each agent will be different, a concept read by one agent will be developed differently to the way in which it will be developed by the agent which sent it.

Each concept resides as an object of class Concept in the Java program, and each object contains all relevant information about the concept, including the information representing the concept and the values calculated to assess it. To communicate a concept, a pointer to the concept is put in the inboxes of all the other agents. Therefore, the receiving agent has access to all the information about the concept. However, to actually read a concept from the inbox, we force the receiving agent to reconstruct the concept from scratch using the construction history of the concept. The disadvantage to this is the additional time spent repeating constructions. However, as each agent may be working with different examples from the theory, the representation of a concept sent by an agent may not convey all the information required by the receiving agent. For example, if one agent was working with the numbers 1 to 10, and received a concept from an agent working with 1 to 5, it would effectively have to construct the concept from scratch to fill in the missing details.

As each agent measures interestingness in a different way, an agent receiving a concept will either have to accept the judgement of the sending agent, or re-assess the concept on its own terms. For a heuristic search to perform correctly, the last option is preferable, and so each agent assesses a communicated concept itself and places it in the appropriate place in the agenda. As some of the measures are built up as the concept is constructed, the simplest way for a concept to be re-assessed correctly is to build it from scratch. Therefore, another advantage to reconstructing concepts is that each is properly assessed and incorporated into the theory correctly.

Each agent passes *all* of its concepts to the other agents. At present, whenever an agent sends its concepts, it takes the opportunity to read the concepts in its inboxes. This model may change in future, as we test whether the agency is more efficient if agents send concepts more often than they read them. Whereas all concepts are sent to inboxes, agents are very selective about which concepts they take from the inboxes. The selection procedure is dependent on the task set for the agency, so we discuss this in the context of the problem being addressed in §4.

There are only a few settings available to the user to fine tune the action of the agency:

- The user can choose how many agents to use.
- They can set the search parameters differently for each agent, so that they perform different searches.
- They can detail how each agent selects from its inboxes - including specifying the agents it takes concepts from and which concepts to take.
- They can specify when the sending and reading of concepts takes place. This is specified in terms of how many theory formation steps occur before the agent communicates the concepts it has found (and reads the concepts in its inboxes).

4 Exhaustive Categorisation

When HR is asked to explore a domain it must find out as much information about that domain as possible. In this mode it is difficult to assess how well the program is doing. Certainly it is impossible to find all the concepts in a domain and even in a depth limited search, there are often too many concepts for a program to conceivably find in a reasonable time limit. Also, it is very difficult to assess how creative a program has been in constructing a particular theory. The exhaustive categorisation problem discussed here provides ways to measure the success and creativity of a theory formation program.

4.1 Problem Description and Motivation

We say that a set of examples has been **exhaustively categorised** by a theory if for any possible way to categorise the examples, there is at least one concept in the theory

which achieves that categorisation. For example, given the integers 1 to 4 as examples for number theory, the entire set of categorisations for these integers is:

$[1,2,3,4]$	$[1,2,3],[4]$	$[1,2,4],[3]$
$[1,2],[3,4]$	$[1,2],[3],[4]$	$[1,3,4],[2]$
$[1,3],[2,4]$	$[1,3],[2],[4]$	$[1,4],[2,3]$
$[1,4],[2],[3]$	$[1],[2,3,4]$	$[1],[2,3],[4]$
$[1],[2,4],[3]$	$[1],[2],[3,4]$	$[1],[2],[3],[4]$

Once the objects of interest supplied by the user have been exhaustively categorised, a milestone has been passed because the program has learned an answer to any question of the form "Why are x, y and z the same but different to a, b and c". For example, if we asked why 1 and 4 are the same, but different to 2 and 3, any program which had invented the notion of square numbers could answer that 1 and 4 are squares but 2 and 3 are not. At present, we have achieved an exhaustive categorisation of the integers 1 to 5 using HR. This leads us to the problem description: to exhaustively categorise the integers 1 to 6.

4.2 Measuring Success

The number of ways of categorising a set of n objects is defined as the nth Bell number, Bell (1934). The Bell numbers are: $1, 2, 5, 15, 52, 203, 877, 4140, \ldots$ Therefore to exhaustively categorise, say, the integers 1 to 5, HR would need to find concepts which categorised them in 52 different ways. The number of categorisations achieved is some measure of the usefulness of the theory formed and hence of the success of the program.

We introduce the following way to compare two theory formation systems:

Suppose systems A and B both start with the same set of examples and perform the same number of theory formation steps. We say that A is more **creative** than B if the theory it has produced has achieved more categorisations of the examples than the theory produced by B.

There have been entire conferences devoted to understanding creativity in humans and machines,[1] and we are still far from an explanation which could be turned into concrete ways to measure the creativity of a computer program. We believe that a program which produces 100 different categorisations of a set of objects in 500 steps is more creative than one which produces only 10, and this is how we choose to compare the creativity of agencies. We certainly do not claim to have captured the very essence of creativity with these measures. Note that because each agent must reconstruct any concept it decides to read, these reconstruction steps count as theory formation steps. Therefore an agency does not get any steps for free, and the creativity measure is valid.

[1] For example the International Congress on Discovery and Creativity, Ghent 1998.

We hypothesise that employing an agency will improve the creativity of HR. We test this hypothesis in experiment 1. In experiment 2, we assess whether we can improve efficiency without losing creativity. In experiment 3 we assess whether the increase in creativity of an agency compensates for the loss of efficiency due to the communication overheads.

4.3 Experiment 1 - Creativity

We used four agents named:

(H)ardy, (R)amanujan, (L)ittlewood and (W)right.[2]

Each agent worked with the numbers 1 to 6, and employed a different search strategy:

- Hardy - Unary first search
- Ramanujan - Depth first search
- Littlewood - Best first search based on novelty only
- Wright - Breadth first search

The only shared resource was the set of 203 categorisations of the numbers 1 to 6 which are calculated beforehand. Each agent removes a categorisation from the set if it is the first to find a concept which achieves that categorisation. The watcher records which agent found each categorisation first.

We experimented with the criteria by which an agent chooses concepts from its inboxes. We first allowed each agent to read every concept produced by every other agent, but as expected, there was so much repetition of work that the agencies fared much worse than HR running alone. A better alternative is to only allow concepts into the theory which are new to the agent. However, the only way to tell that a concept is new is to test the datatable against all those already in the theory, which is time consuming.

Finally, we realised that for this problem, the most natural selection procedure is to only reconstruct concepts which produce a categorisation which is novel *for the receiving agent.* Because all information about a concept is available, and the agents are working with the same set of examples, it is very quick to check whether a concept's categorisation has already been found by an agent. Further, choosing concepts with novel categorisations guarantees the novelty of the concept itself, and combination with other concepts in the theory is likely to lead to yet more novel categorisations.

To test our hypothesis we compared the creativity of every agency possible using the four agents. These included all agencies with one agent, named H, R, L and W, all agencies with two agents, named HR, HL, HW, RL, RW and LW, all agencies with three agents, named HRL, HRW, HLW and RLW and the agency with four agents, named HRLW. For each of the agencies with two or more agents, we tested two copies: one with no communication at all, and one with **immediate communication** - ie. after every theory formation step, each agent reported

[2] Four highly collaborative number theorists.

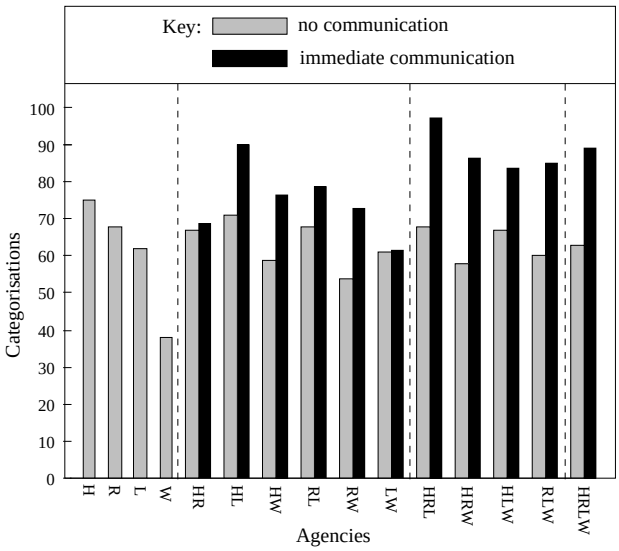

Figure 2: Number of categorisations achieved by agencies after 3000 theory formation steps

new concepts and read those reported by others. We ran each agency for a total of 3000 theory formation steps. In agencies of n agents, each was allowed to perform only $3000/n$ steps. So, in agency HRL, agent H performed 1000 steps as did R and L. We chose 3000 steps because it takes around a minute to perform this number and because 3000 is perfectly divisible by 2, 3 and 4. The search was depth limited to complexity 6, and we ran all tests on a Pentium 500Mhz processor.

Before detailing the results from the test, we report an unexpected phenomenon which occurred when running agencies which communicate. We noticed that the number of categorisations being achieved differed when the program was run with exactly the same settings for the same number of theory formation steps. We are still investigating this, and at present we believe that the Java thread mechanism cannot be trusted to perform exactly the same tasks in the same order. This is a problem because agent L uses a best first search by measuring concepts in relation to the others in the theory. Suppose that agent L read concept C just before it was going to invent a very interesting concept of its own, X. If C was interesting, it would be developed due to the best first nature of the search. Only after C had been developed would X be formed, which leaves less time to develop it. In a different session, if L read concept C just after it invented X, X would be developed before C, giving more time to develop X and the concepts produced from it.

Thus, because our sessions are limited by the number of steps allowed, small changes in the timing of the communication of concepts can make substantial differences in the theories produced. Without explicitly introducing stochastic processes, this models to some extent the way in which luck and serendipity can influence the development of mathematical theories. Imagine the advances which would have been made if Fourier had not lost Galois' manuscript and had saved him from the fatal

	expected number of categorisations	
agency size	no communication	immediate communication
1	60.75	n/a
2	63.3	74.75
3	63.25	88.15
4	63	89.2

Table 1: Expected number of categorisations for agencies of different sizes after 3000 steps

duel.[3] However interesting the phenomenon is, it makes testing difficult, and we were forced to average the results over 10 sessions to compensate for the difference in theories produced when using a communicating agency.

Figure 2 shows the number of categorisations found by each agency, with the grey boxes for agencies with no communication, the black boxes for agencies with immediate communication. These results are fairly conclusive. In only two immediate communicating agencies containing a particular agent did the agency containing just that agent perform better (agency HW performed worse than H, and LW performed worse than L).

If we average the scores over agencies of the same size, we can look at the expected number of categorisations for an agency of a given size. As portrayed in table 1, the most creative agency is the immediate communicating agency with 4 agents, which slightly outperforms the average immediate communicating agency with 3 agents. It is also clear that the increase in creativity is due to the communication, not just the fact that the system is using multiple search strategies which cover different areas of the space. In every case the agency with communication outperformed the agency without communication.

4.4 Experiment 2 - Communication Intervals

One way to improve efficiency is to reduce the number of times concepts are communicated and inboxes are checked. With the agencies discussed above, after every theory formation step, if a new concept had been produced it was communicated to the other agents. Similarly, after each step, every agent checked their inboxes for new concepts, and read any which produced a new categorisation. Delaying the reading and communicating of concepts until after, say, every 10th step, will reduce some of the communication overheads. We tested whether delaying the communication would affect the creativity of the agency.

We took the best agency from the first experiment, HRL, and ran it for 100 theory formation sessions. We increased the waiting time for communication from 1 to 10 to 20, etc. up to 1000 steps and recorded the number of categorisations it produced after 3000 steps. Again agents H, R and L each performed 1000 steps. We also recorded the number of categorisations which were **multi-**

[3] See Stewart (1989) for more details of this tragedy.

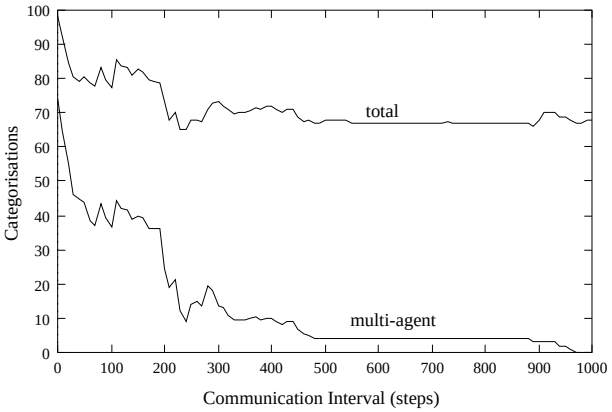

Figure 3: Effect of communication intervals on HRL agency and correlation with multi-agent categorisations

agent. A categorisation is defined to be multi-agent if the first concept which achieved the categorisation had concepts from more than one agent in its construction history. As before, we repeated this experiment 10 times and took an average to counteract the phenomenon described above. Figure 3 shows the effect of lengthening the communication intervals on the total number of categorisations formed and the number of multi-agent categorisations formed.

From figure 3, we see that increasing the communication interval will in general decrease the creativity of the system, and that the total communication scheme outperforms the others significantly. The decrease in creativity is not smooth, however, and we are presently studying the theories produced to explain the peaks and troughs observed in this experiment. The correlation between the total number of categorisations and the number of multi agent categorisations was more pronounced than we expected. Judging by the correspondence in the peaks and troughs on the two graphs, if an opportunity to find multi-agent categorisations is missed, this is not compensated by increased time spent by agents searching on their own.

The decrease in quantity of categorisations is due to our limitation of only 1000 theory formation steps for each agent. For example, agents in the agency where communication occurs after the 600th step only communicate their concepts once, and only those invented before the 600th step. Any interesting concepts it finds after the 600th step are never communicated. This explains the long horizontal sections of the graphs in figure 3 - concepts are communicated too late to be used sufficiently to find multi-agent categorisations. As we see by the end of the total curve, late introduction of the concepts actually hinders the creativity of the agency, and when they are introduced too late to be developed at all, the number of categorisations increases.

The correlation between the number of categorisations and the number of multi-agent categorisations, coupled with the reduction in the number of concepts communicated explains why the creativity of the system declines as the communication interval increases. Therefore we hypothesise that if a system only has a limited number of steps to perform, the smaller the communication interval, the more creative the system will be. More experimentation is required in different theories and with more theory formation steps to confirm this hypothesis.

agency size	expected number of categorisations
1	96
2	114.7
3	110.35
4	116.6

Table 2: Expected number of categorisations for agencies of different sizes after 5 minutes

4.5 Experiment 3 - Efficiency

Having determined which agencies are the most creative, we must assess whether this increases the overall efficiency of the system - the ultimate aim of an agency. Due to the increased overhead in communicating concepts, it may be that agencies can produce more categorisations in a given number of steps, but it takes them longer to carry out those steps because the communication slows down the process. Of course, it is certainly possible to run each agent on a different processor which will give efficiency gains more than compensating for the increased communication overhead. However, it was useful to test the efficiency of the system as a whole.

We scaled the problem up by running each single agent agency and each total communicating agency for a duration of five minutes and comparing the number of categorisations produced. As before, we averaged the results for the communicating agencies over 10 sessions. Finally, we averaged the results over agencies of each size and recorded the results in table 2. We see that the multi-agent agencies are more efficient in general, and we hypothesise that the additional overhead is compensated by the increase in creativity. Again, we plan more experimentation to further investigate this hypothesis.

4.6 Utility and Clarity

To end our investigation of the exhaustive categorisation problem, we looked at the utility of each agent - whether it found any novel categorisation before the other agents. We found that in every run, each agent contributed at least one categorisation to the theory. Figure 4 gives the proportion of categorisations which were first introduced by each agent for a sample 3000 step session using the HRLW agency. It also details the proportion of the categorisations introduced first by each agent which were multi-agent. We see that agent H introduces half of the categorisations. This is due to the effectiveness of its search strategy, and not because H collaborated the most, as we see from the second pie chart that the number of multi-agent categorisations introduced by the agents was roughly equal.

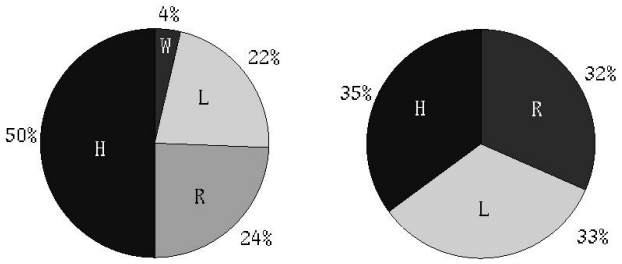

Figure 4: Proportion of (i) all categorisations and (ii) multi-agent categorisations found by agents in an example session with agency HRLW

Agent W does not perform well on this problem and it seems clear that some aspect of a depth first search needs to be incorporated. Also, agent W puts concepts it reads from other agents to the bottom of the agenda, which explains why it produces no multi-agent categorisations. We affectionately call agent W the pedantic agent, as it attempts to cover all possibilities thoroughly while its colleagues race off in many directions. However, agent W performs an important function: it improves the clarity of the theories produced. We can measure the **clarity** of a theory as the average number of theory formation steps required to form a concept from the theory.

A breadth first search will produce the simplest theories, as it will not build concepts of complexity three before building all concepts of complexity two and so on. More than this, as discussed in §2, if a concept is found which matches one already in the theory and the new concept has a more concise definition, the simpler definition is kept. In this way, if agent W reads a concept from another agent and later finds a more concise definition, the theory will benefit from the pedantic approach as clearer definitions for complicated concepts will be produced. We have not compiled the statistics for the gain in clarity of theories obtained when agent W is in the agency, but intend to do so when we run further tests on these agencies.

5 Further Work

Improvements in efficiency could be made by sharing parts of the agenda between agents, because presently an agent sending a concept and an agent receiving the concept will develop it in some identical ways. We also hope to use agencies to improve the modelling of cross domain theory formation as discussed in Steel et al. (2000). At present, to produce cross domain concepts, HR must encourage the combination of two concepts from different domains. Considerations also have to be made in the assessment of the cross domain concepts. We anticipate that using an agent to produce a theory in each different domain, with concepts being communicated between agents and hence across domains will greatly improve the model for cross domain theory formation.

5.1 Extrapolating Integer Sequences

As described in Colton et al. (2000a), HR has been adapted to perform theory formation in a goal based way in order to perform machine learning tasks. It is used to find a concept with examples which match the examples given by the user. This can be applied to the problem of extrapolating integer sequences, where the concept to be learned is a sequence. HR performs a forward chaining search and is equipped with a lookahead mechanism enabling it to spot when it has found the concepts necessary for the definition. For example, given the sequence of odd primes, as soon as HR invents odd numbers and then prime numbers, it looks ahead and notices that their combination will produce the desired concept.

We are currently experimenting with an agency approach to sequence extrapolation, motivated by limitations when extrapolating certain sequences. These sequences highlight some of the difficulties HR faces:

What is the next in these sequences?

- 2,3,5,7,11
- 1,3,6,11,18,29
- 101,102,104,106,110

The first is identified by HR as the sequence of prime numbers. The other sequences cause more difficulty.

If we calculate the difference between successive terms of the second sequence, we get the prime numbers again: 2, 3, 5, 7, 11. In IQ tests where sequence extrapolation problems are common, knowledge of this difference transformation is expected. This transformation is so common that it suggests tailoring HR's forward looking mechanism to look for either the original sequence or the difference sequence. This caused many technical problems, and was a messy solution. Moreover, in future versions of HR we hope to make the search more goal directed, using the examples supplied to explicitly direct the search. In this case, as the original and difference sequence are often so different there will be a conflict in the search strategies. Therefore, we are experimenting with a multi-agent approach to sequence learning, where two agents are employed, one to look for the original sequence and the other to look for the difference sequence. The model is certainly much neater and works well. We are still testing whether multi-agent model is more efficient than the single-agent model, but initial findings are encouraging.

If we now look at the third sequence and take 99 from each term, we get: 2, 3, 5, 7, 11, which is the prime sequence for the third time. This is not, however, a common transformation, and there are too many similar transformation to reasonably dedicate an agent to the output from each one. A better model is to take each concept produced and determine which, if any, transformation from a family of transformations would produce the desired sequence. For example, when HR invents the concept of prime numbers, it could look at the family of transformations which add on a particular number to every term in

the sequence. To get from the first term of the prime sequence it has just invented to the first term of the target sequence, it needs to add on 99. It would then determine that adding 99 is the desired transformation.

We are presently testing whether it is more efficient to have a single agent attempting to identify the correct transformation. This agent takes the concepts from the other agents which could possibly be transformed and attempts to find the correct transformation. Again, the model works well, and we are assessing whether this is efficient. We cannot possibly hope to cover all possible sequences which an inventive person could produce, but we do hope to show that the agent based approach improves the coverage of the system.

6 Conclusions

Adapting HR to employ a multi-agent approach is a natural way to extend its theory formation abilities. Agencies equipped with a method for communication of concepts and selection of concepts better model the way in which collaborative research progresses in science than single programs running in isolation. In Furse (1990), the author proposes a network of theory formation programs each communicating their most interesting concepts to the programs on the network. This is a good model, and similar to the one we have implemented. However, in our model, each agent communicates all its concepts to the others, but will assess a concept on its own terms rather than accepting the assessment of the sending agent. In this way, a concept which seems dull to one agent may be picked up and fruitfully utilised by another.

Machine learning programs such as Progol, Muggleton (1995), are asked to find a single concept which classifies the examples supplied correctly. Therefore, among other ways, success can be measured by the number of concepts it can learn from a predefined set. It is more difficult to perform a quantitative assessment of a theory formation program, because the goal is to find many interesting concepts and obtain some understanding of the domain. We have chosen to measure the number of different ways a set of examples can be categorised by a theory to determine the quality of the theory and accordingly developed a way to compare the creativity of different systems.

We have demonstrated that a multi-agent approach can increase the creativity and efficiency of a system, even before any advantage is gained from running each agent in parallel. Autonomous intelligent agents have emerged over the last decade as an important technique to solve many interesting problems, and improve efficiency in many areas, Jennings and Wooldridge (1997). We have shown that theory formation programs can benefit from an agent based approach. We also hope to demonstrate that the theory formation agencies we plan to implement in the future will apply fruitfully to other areas of artificial intelligence such as machine learning and theorem proving.

Acknowledgements

We wish to thank Edmund Furse for in depth and informed discussions about possibilities for multi-agent theory formation. We would also like to thank the anonymous reviewers for their enthusiastic comments about the extended abstract for this paper, and the symposium chair, Geraint Wiggins, for his effort in bringing us all together. This work is supported by EPSRC grant GR/M98012.

References

E Bell. Exponential numbers. *American Mathematics Monthly*, 41:411–419, 1934.

S Colton, A Bundy, and T Walsh. HR: Automatic concept formation in pure mathematics. In *Proceedings of the 16th IJCAI*, 1999.

S Colton, A Bundy, and T Walsh. Automatic identification of mathematical concepts. In *Machine Learning: Proc. of the 17th International. Conference*, 2000a.

S Colton, A Bundy, and T Walsh. On the notion of interestingness in automated mathematical discovery. *IJHCS*, Forthcoming, 2000b.

R Davis and D Lenat. *Knowledge-Based Systems in Artificial Intelligence*. McGraw-Hill Advanced Computer Science Series, 1982.

S Epstein. Learning and discovery: One system's search for mathematical knowledge. *Computational Intelligence*, 4(1):42–53, 1988.

E Furse. Why did AM run out of steam? Technical Report CS-90-4, Department of Computer Studies, University of Glamorgan, 1990.

N Jennings and M Wooldridge. *Agent Technology: Foundations, Applications and Markets*. Springer, 1997.

T Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1970.

D Lenat. Eurisko: A program which learns new heuristics and domain concepts. *Artificial Intelligence*, 21, 1983.

S Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

M Sims and J Bresina. Discovering mathematical operator definitions. In *Machine Learning: Proc. of the 6th International Conference*. Morgan Kaufmann, 1989.

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.

I Stewart. *Galois Theory*. Chapman and Hall Mathematics, 1989.