Automatic Generation of Some Results in Finite Algebra

Masayuki Fujita[1], John Slaney[2] and Frank Bennett[3]

Mitsubishi Research Institute[1]
Arco Tower, 8-1
Shimomeguro, 1-chome
Meguro-ku, Tokyo 153, Japan

Automated Reasoning Project[2]
Australian National University
GPO Box 4, Canberra, ACT
Australia

Department of Mathematics[3]
Mount St Vincent University
Halifax, Nova Scotia, Canada

Abstract

This is a report of the application of the Model Generation Theorem Prover developed at ICOT to problems in the theory of finite quasigroups. Several of the problems were previously open. In this paper, we discuss our theorem proving methods, related to those of the existing provers SATCHMO (Manthey, Bry) and OTTER (McCune), and note how parallel processing on the ICOT Parallel Inference Machines was used to obtain high speeds. We then present and discuss our machine-aided investigation of seven problems concerning the existence of types of quasigroup.

The field of finite algebra is rich in problems suitable for computational treatment. In particular, questions about the existence of structures of given finite sizes may usefully be approached using techniques for theorem proving or for constraint satisfaction. Perhaps the best known recent result of this sort is that of Lam, Thiel and Swiercz [Lam *et al,* 1989] showing that there is no projective plane of order 10. The present paper reports further research in the same tradition in which our tool was the ICOT theorem prover MGTP and in which we were able to obtain new existence and nonexistence theorems for certain interesting classes of quasigroup. It should be noted that MGTP is a general-purpose theorem prover not at all designed with quasigroups in mind. Hence this application of Artificial Intelligence research to open mathematical problems was rather unexpected. We welcome it, however, as we feel that automated reasoning has reached the point where it can and should address the needs of researchers in other disciplines.

1 Finite domain searching

Where a first order theory has finite models, a reasonable way to find some is to fix on a particular finite domain of objects and to search for interpretations of the function and predicate symbols as functions and predicates on that domain, constrained to make all of the axioms of the theory true. Finite model generation may thus become a constraint satisfaction problem of the familiar sort, and may yield to the familiar techniques approprite to such problems. To illustrate with a deliberately trivial example, consider the theory of semigroups. This has only one axiom, an equation

$$a \circ (b \circ c) = (a \circ b) \circ c$$

and has, of course, models of every cardinality. To find the semigroups of order 3 (for instance) we fix the domain as consisting of three objects; the numbers 1, 2 and 3 will do nicely. Now we need to determine the value of $x \circ y$ for each choice of x and y from the domain, giving 9 cases to determine and 3 possible values for each case. Evidently there are 3^9 or 19683 possible vectors of 9 values in this search space. The "good" ones are those which make true every ground instance of the associativity axiom such as $2 \circ (1 \circ 3) = (2 \circ 1) \circ 3$. So each bad vector contains a set of (at most) four values which suffice to refute some such formula. For example, the four assignments

$$\begin{aligned} 1 \circ 3 &= 2 \\ 2 \circ 1 &= 2 \\ 2 \circ 2 &= 3 \\ 2 \circ 3 &= 1 \end{aligned}$$

together force $2 \circ (1 \circ 3) \neq (2 \circ 1) \circ 3$, thus violating associativity. Therefore the disjunction

$$(1 \circ 3 \neq 2) \vee (2 \circ 1 \neq 2) \vee (2 \circ 2 \neq 3) \vee (2 \circ 3 \neq 1)$$

is one of the ground negative constraints of cardinality 4 which all good vectors must obey.

There are many methods of enumerating the vectors which do obey such a set of constraints. We used a backtracking search technique based on automated deduction with a clausal representation of the problem, as detailed below, but there is no reason why other styles of search such as that embodied in arc consistency or path consistency algorithms should not be employed to much the same effect. Our concern in this paper is to report the method which was successful in practice, not to prove that it is the best method possible.

Whatever techniques are used, some general heuristics are in order. Firstly, whenever one of the ground negative constraints, as instanced above, is used to refute an attempt to build a model, it should be remembered somehow and the search control should ensure that it

never again gets incorporated into a partial model candidate. The search will typically backtrack many times, but it should never do so twice for the same reason. Secondly, it obviously pays to minimize the furcation of the search tree. Hence where there is a choice as to which cell is to be given a value next, choose one with the smallest number of possible values (given the partial structure already in place). An early version of our program MGTP failed to do this and constructed over 52 million branches on problem QG5.7 below; after the heuristic was added, it branched just 9 ways on the same problem! Thirdly, as is well known, most algebraic search problems permit early detection of some or all isomorphisms and by attending to these we may frequently cut down the search space by some orders of magnitude. In the cases treated below, we used a simple initial restriction of the problem to avoid searching a great many isomorphic subspaces.

Naturally, we do not claim originality for any of these general search heuristics. Indeed, they are rather wellworn. Nonetheless, we wish to draw attention to them since they were essential to the success of our experiments and since it is still common to find such obvious points overlooked.

2 The program: MGTP

2.1 Otter, Satchmo and MGTP

Otter [McCune, 1990] is a very efficient first order theorem prover. It can be seen as computing the closure of a set of axioms under selected rules of inference. It works with two set of clauses called the Usable Set (U) and set of support (SOS). In each step it moves a clause C from SOS to U, generates immediate consequences of C in combination with with members of U and stores them in SOS. Essential to its strategy is the avoidance of possible regeneration of redundant consequences mainly by rejecting subsumable clauses instead of storing them (forward subsumption). It may also apply backward subsumption, whereby newly kept clauses are used to simplify the existing U and SOS. It gives the option of various rules including resolution and some of its more powerful relatives such as hyperresolution and unit-resulting resolution, as well as a wide variety of equality reasoning facilities such as forms of paramodulation and demodulation.

Satchmo [Manthy and Bry, 1988] can be seen as a specialized theorem prover for solving only range restricted problems, using the important technique of case splitting. Range restricted problems are those which assure that all derived positive clauses are ground. Case splitting, as is familiar, is a matter of treating clauses of a certain type (usually, as in Satchmo, positive ground clauses) by assuming each of their literals in turn in order to reason by cases. It is thus fundamental to reasoning in the style of semantic tableaux. Given range restrictedness, case splitting is safe from the problem of common variable handling between the split literals. Like the Prolog Technology Theorem Prover described in [Stickel, 1988] Satchmo takes advantage of Prolog's optimization techniques by compiling clauses for runtime efficiency.

MGTP (Model Generation Theorem Prover) is written in the parallel logic programming language KL1. There are two versions of the program: MGTP/G for range-restricted problems and MGTP/N for (Horn) nonground problems. See [Fujita *tt a*/, 1992] for a description. The basic algorithm of MGTP/G is equivalent to that of Satchmo, while that of MGTP/N is based on that of Otter. Both versions run sequentially on the PSI workstations and also on the PIM (Parallel Inference Machine) developed at IGOT. For the algebraic problems considered in the present paper, only MGTP/G was required. Satchmo's technique of compiling clauses into a logic programming language is adopted by MGTP naturally but not trivially. How MGTP/G uses the power of KL1 to maintain efficiency [Fuchi, 1990; Fujita and Hasegawa, 199l]is outside the scope of this paper. This basic efficiency, is the one of the important sources of MGTP's success, even though without heuristics there is no hope of managing the combinatorial explosion.

2.2 Problem representation and heuristics

As in most artificial intelligence applications, heuristics appear to be very important in the attack on difficult model generation problems. All advanced uses of Otter, such as solving the very difficult condensed detachment and related problems reported in [Wos *et a*/, 1990], use weighting mechanisms for picking the next clause from SOS. The simplest method of assigning weights to clauses is to count the constituent literals, and the next simplest is to count symbols. The latter is Otter's default. More elaborate weighting techniques are of considerable interest, but are not the focus of the present paper.

Heuristics for MGTP are in a sense similar to those for the Otter. We used a weighting function of the number of literals in each clause, corresponding to the number of ways the search tree will branch. A unit clause has lowest weight of all. We then simply choose to split a clause of lower weight rather than one of higher. Clearly, this is an implementation of the general principle alluded to in the last section, of minimising furcation of the search tree. In counting literals for this purpose, we omit any that could directly produce the empty clause by clashing with a negative clause. We also omit any literal L such that there exist clauses

$$\begin{aligned} &\mathbf{false} \mathbin{:-} L, a_1, \ldots, a_n. \\ &a_1 \mathbin{:-} \mathbf{true}. \\ &\vdots \\ &a_n \mathbin{:-} \mathbf{true}. \end{aligned}$$

in the set U, since we can look far enough ahead to see that such L could be resolved away and will therefore generate only a dead branch.

Of course such weighting methods may destroy completeness if the domain of the problem is infinite, but there are many ways to escape from this. For example, Otter has the strategy of moderating its weight-directed search by letting every nth clause come from a breadth first search. For finite domain problems such as n-queen problems and the Peiletier and Rudnicki problems, however, a weight strategy such as ours does not endanger completeness. Moreover it prunes an extremely large

```
dom(1), dom(2), dom(3), dom(4) :- true.

p(M,N,1); p(M,N,2); p(M,N,3); p(M,N,4)
    :- dom(M), dom(N).

false :- p(X,4,Y), {Y+1<X}.

false :- p(X,X,U), {X≠U}.

false :- p(X,Y,U), p(X,Y1,U), {Y≠Y1}.

false :- p(X,Y,U), p(X1,Y,U), {X≠X1}.

false :-
    p(E,X,Y), p(Y,E,Z), p(Z,E,U), {X≠U}.
```

Figure 1: MGTP input for QG5, order 4

percentage of the branches in all of the finite quasigroup problems.

The problem description for MGTP theorem prover is very simple (Figure 1). The search space of this naive representation is 4^{16} branches. However the heuristics above succeed in reducing it to just a single branch.

2.3 Parallelization

Even though the heuristics dramatically cut down of search space, the hardest problem we solved had millions of cases in the search space as shown in section 3.2. Such search can be split to several processes at any stage of case splitting, since only ground clauses are thus treated and hence no process needs to communicate with any other in order to solve its case. The OR-parallelism natural to this problem is enough to exploit almost linear speed-up with 256 processors on the PIM-m machine developed at ICOT. PIM-m is a 2-dimensional square mesh MIMD machine capable of up to 160 M append LIPS. We solved the hardest problem with 2,749,676 branches in a little under 4 hours by this machine. It would require over 30 days by a single processor.

For load distribution, a simple stochastic mapping function chooses relatively distant processors. The level of task distribution of these classes of problem has been shown to be close to optimal (Figure 2), so we did not implement a further level of dynamic load balancing by observing and communicating each processor's load level. This achievement was based on a MIMD machine with fine grain and low communication cost; on less advanced MIMD computers it would not be so easy.

3 Problems and results

3.1 Some quasigroup existence problems

A quasigroup [Denes and Keedwell, 1974] is a set on which is defined a binary operation • such that the equations $a \cdot x = b$ and $y \cdot a = b$ have unique solutions for all elements a and b. Evidently, the multiplication table of a quasigroup defines a Latin square. A quasigroup (like any other algebraic structure) is *idempotent*

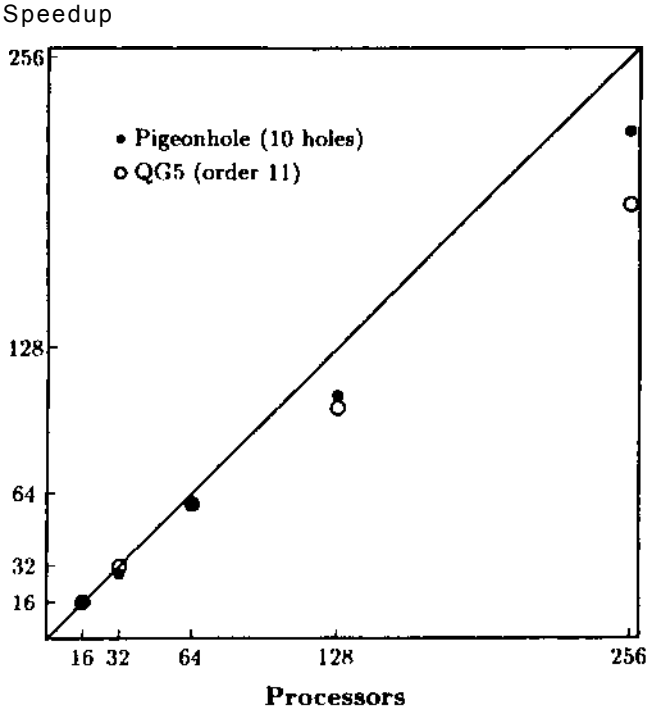

Figure 2: Speedup on two problems on PIM-m

iff $a \cdot a = a$ for every element a. For the purposes of this paper, we are interested in two types of question regarding finite quasigroups: the existence of conjugate-orthogonal [idempotent] quasigroups of certain orders, and the spectra of certain quasigroup equations.

Where $(Q,\cdot)$ is any quasigroup, we may define on Q some further operations o_{ijk} where i, j and k are distinct members of $\{1,2,3\}$.

$$
\begin{aligned}
x \circ_{123} y = z &\iff x \cdot y = z \\
x \circ_{213} y = z &\iff y \cdot x = z \\
x \circ_{132} y = z &\iff x \cdot z = y \\
x \circ_{312} y = z &\iff y \cdot z = x \\
x \circ_{231} y = z &\iff z \cdot x = y \\
x \circ_{321} y = z &\iff z \cdot y = x
\end{aligned}
$$

Then each $(Q, \circ_{ijk})$ is a quasigroup, called a *conjugate* of $(Q,\cdot)$. We shall refer to $(Q, \circ_{ijk})$ as the (i,j,k)-conjugate of $(Q,\cdot)$.

Orthogonality is defined in a standard way. Let $(Q,\cdot)$ and $(Q,\star)$ be two quasigroups defined over the same set Q. They are orthogonal iff $x \cdot y = z \cdot t$ and $x \star y = z \star t$ together imply that $x = z$ and $y = t$. A quasigroup which is orthogonal to its (i,j,k)-conjugate is said to be (i,j,k)-conjugate orthogonal. One which is $(2,1,3)$-conjugate orthogonal is more commonly said to be self-orthogonal. The acronyms COLS and COILS abbreviate 'conjugate-orthogonal Latin square' and 'conjugate-orthogonal idempotent Latin square' respectively, and the expressions (i,j,k)COLS(v) and (i,j,k)COILS(v) denote respectively an (i,j,k)COLS of order v and an (i,j,k)COILS of order v.

We have investigated the following seven specific problems from [Bennett and Zhu, 1992]. The bold numbers in parentheses are those given to the open cases of these problems on pages 88-90 of [Bennett and Zhu, 1992].

1. Establish the existence or nonexistence of a (3,2,1)-COILS(v). (1)
2. Establish the existence or nonexistence of a (3,1,2)-COILS(i;). (2)
3. Find a Schroder quasigroup of order n. In particular, find an idempotent one. (18)
4. Find a quasigroup of order n satisfying Stein's third law yx.xy — x. In particular, find an idempotent model. (19)
5. Investigate the spectrum of [idempotent] quasigroups satisfying the identity (yx.y)y = x. (22)
6. Investigate the spectrum of quasigroups satisfying the identity xy.y — x.xy. Is it restricted to n = 0 or 1 (mod 4)? (23)
7. Investigate the spectrum of quasigroups satisfying the identity yx.y = x.yx. Is it restricted to n = 1 (mod 4)? (24)

Some brief explanations and comments are in order. For more detail on all of these problems, the reader is referred to [Bennett and Zhu, 1992].

1. The open problem is the case v = 12. Otherwise, it is known that solutions exist for every positive integer except for 2, 3 and 6.
2. The problem of existence of (3,l,2)-COILS(v) has almost been solved. There are only four orders left undecided, namely, v = 10, 12, 14 and 15. Solutions exist for every other positive integer except for 2, 3, 4 and 6.
3. The identity xy.yx — x is known as Schroder's second law, and quasigroups satisfying it are called Schroder quasigroups. They are well known to be self-orthogonal. The spectrum of Schroder quasigroups contains precisely the set of all positive integers n = 0 or 1 (mod 4) except n — 5 and possibly excepting n — 12. Interestingly, the idempotent Schroder quasigroups have the same spectrum, with the additional exception of order 9.
4. The identity yx.xy = x (Stein's third law) also forces quasigroups to be self-orthogonal. Its spectrum contains precisely the set of all positive integers n = 0 or 1 (mod 4) except possibly n = 12. Moreover, there exists an idempotent model of each such order except 4 and 8, and possibly 12.
5. The spectrum of the identity {yx.y)y - x was investigated in detail in [Bennett, 1989]. Before the present research, it was known to contain every integer $n > 1$ with the exception of 2, 6 and possibly excepting $n \in \{10,14,18,26,30,38,42,158\}$. The existence of idempotent models is in rather more doubt. The known exceptions listed in [Bennett, 1989] are 2, 3, 4 and 6; there are 56 undecided cases, the smallest being 9, 10 and 12-16.
6. The spectrum of the identity xy.y = x.xy (sometimes called Schroder's first law) is not very precisely known. In [Bennett and Zhu, 1992] we report that it contains all integers n > 1, where n = 0 or 1 (mod 4), with the exception of n. = 5 and a list of 35 possible exceptions the smallest of which are 9 and 12 and the largest 177. We do not know whether the spectrum is restricted to n = 0 or 1 (mod 4). Note that all quasigroup models of this identity are idempotent.
7. The spectrum of the identity yx.y = x.yx contains all positive integers n = 1 (mod 4) except possibly for n = 33. It remains an open problem to determine it more precisely. Is it restricted to n = 1 (mod 4)? Again, all models are idempotent.

3.2 Results

We present the results generated by MGTP for the seven problems detailed above. In each case we searched for quasigroups of given orders either showing that there exists no model or generating such of the models as were within the search space. Some, but not all, isomorphic copies of models were omitted as a result of the search strategy. As stated in table 3, twelve of the cases were left as open problems in [Bennett and Zhu, 1992]. The order 9 case of problem QG5 had previously been solved by Zhang [Zhang, 199l] and by us [Slaney, 1992].

In summary, the conditions defining these problems are as follows.

QG1 If $ab = cd$ and $a \circ_{321} b = c \circ_{321} d$ then $a = c$ and $b = d$

QG2 If $ab = cd$ and $a \circ_{312} b = c \circ_{312} d$ then $a = c$ and $b = d$

QG3 $ab \cdot ba = a$

QG4 $ba \cdot ab = a$

QG5 $(ba \cdot b)b = a$

QG6 $ab \cdot b = a \cdot ab$

QG7 $ba \cdot b = a \cdot ba$

In most cases the quasigroups were also stipulated to be idempotent:

$$x \bullet x = x$$

The elements of the domain for all models were taken to be the natural numbers 1 ... M where M is the order of the desired quasigroups. In order to avoid searching certain subspaces isomorphic to those already searched, we adopted the additional axiom

$$a \blacksquare M \geq a - 1$$

Clearly this sacrifices no generality, and eliminates many isomorphic copies though not all. There are more effective ways of cutting out automorphs, which are the subject of some current and projected research, but we present the results of this section as evidence that our rough method works well enough for some non-trivial purposes.

Figure 3 shows in each case the total number of backtracks (models plus failure branches) in the search tree, the number of solutions found and the time in seconds

Problem	Order	Branches	Models	CPU Time (seconds)
QG1	8	180,430	16	1894
QG2	7	1,114	14	48
QG3	7	183	0	6
	8	3,875	18	28
	9	312,321	0	1022
QG4	7	123	0	6
	8	3516	0	23
	9	314,925	178	1127
QG5	7	9	3	3
	8	34	1	7
	*9	239	0	12
	*10	7,026	0	66
	11	51,904	5	224
	*12	2,749,676	0	13715
	†*10	4,473,508	0	13101
QG6	*7	7	0	2
	8	20	2	6
	*9	160	4	14
	*10	2,881	0	43
	*11	50,888	0	248
	*12	2,420,467	0	8300
QG7	*7	182	0	4
	*8	160	0	5
	9	37,026	1	90
	*10	1,451,992	0	2809

* Previously open problem
\+ Without assumption of idempotence

Figure 3: The Results

on PIM-m with 256 processors. Several features of the problems are apparent from the table of results.

- The first problems we attempted were QG5 and QG6, for each of which we obtained new results as shown. It is clear from the branching factors and runtimes that these two are very similar in degree of difficulty for our program and that their complexity appears exponential. Hence we feel that the order 13 cases are not feasible at present without some further insight into either the algebra or the algorithm. Similarly, QG3 and QG4 seem to march together in degree of difficulty.
- Problems QGI and QG2 are especially difficult for our program at present. In neither case were we able to get close to the open cases.
- Not all of the previously open problems were particularly hard. The clearest example is problem QG6.7, though the same is true of the small cases of problems QG5, and QG7. This reflects the fact that most previous approaches had been by human mathematicians using analytic reasoning, whereas sometimes a machine aided exhaustive search is more effective.
- Where the result of the search is positive, a model being found, it yields the most satisfying kind of constructive proof, but where it is negative it yields no readily surveyable proof in the difficult cases. Its only report is that it looked everywhere but found no model. While we feel that this is acceptable as a proof,we also acknowledge that an analytic proof of the same result would be welcome. At the least, it would provide some deeper and less case-ridden reason to hold, for instance, that there is no quasigroup of order 10 in which $(yx.y)y = x$.
- The use of an advanced computer was essential to our success in the most difficult cases. The PIM-M has 256 processors, so any search with fewer than 256 branches is too trivial to keep it fully occupied. This accounts for the fact that in some cases the increase in time as the problem size increases is less than the increase in the number of branches. Where the problem size was sufficient to make parallelization worthwhile, however, almost linear speedup was achieved, since the OR-parallel computation requires almost no inter-process communication.

The new theorems are likely to be useful in research on the motivating problems from design theory [Bennett, 1989]. Already we have been able to use the positive result of QG6.9, that 9 belongs to the spectrum of the equation $xy.y — x.xy$, to improve the results of Theorem 7.28 in [Bennett and Zhu, 1992]. From the list of possible exceptions, we can now show that all of 33, 45, 65, 68, 72, 81, 89, 105, 108, 117, 129, 153, 156, 168 and 177 are in the spectrum. Combining this with the computer-generated results, we now know that the latter does not contain 2, 3, 5, 6, 7, 10, 11, 12 but that it does contain all other values of $n = 0$ or $n = 1 \pmod 4$ with the possible exceptions of 17, 20, 21, 24, 41, 44, 48, 53, 60, 69, 77, 93, 96, 101, 161, 164 and 173. The negative results are harder to use in this way, though they do help to focus the search for exceptions.

We wish to remark that the order 12 case of problem QGI would be the most interesting to see solved, since a solution would confirm or remove the only remaining undecided order for the existence of a (3,2,1)-COILS(?;), which is an extremely simple and natural construction. The order 12 cases of problems QG3 and QG4 would also be interesting, since a solution to either problem would similarly complete its spectrum. With respect to these problems, however, order 12 is too large for our current technique, as may be seen from the results in table 3, and so these problems remain open and just out of reach. The order 10 cases of QG3 and QG4 could probably be done, given many hours of computation time on PIM-m, but there would be rather little point to such an exercise since the results are known.

4 Appendix: Further Research

Since the above was written, we have continued to work on the seven problems described in this paper and some cognate ones. In this we have been joined by Mark Stickel of SRI. The full results of our more recent researches will be published in due course in the mathematical literature, and the computational aspects described more fully in a longer paper (in preparation) but a summary should be appended here.

We have used three programs: MGTP, our general constraint satisfaction program FINDER [Slaney, 1992] and an implementation of the Davis-Putnam algorithm for solution of ground satisfiability problems. Each of these programs has now been successful in solving open problems concerning finite quasigroups. The most important of the recent results are positive solutions to QG3.12 and QG4.12. In each case, idempotent structures were discovered. These complete the spectrum for QG3 and QG4, and of course also for the associated classes of block designs and other combinatorial structures as detailed in [Bennett and Zhu, 1992]. We are far from being able to exhaust the search space for order 12 of these problems in a reasonable time, but we have been able to see far enough into it to solve the existence problems.

Another construction of value in determining the spectra of classes of finite COILS is that of an incomplete model. This is a quasigroup with a smaller subquasigroup missing. The postulates such as idempotence and any special equations are required to be satisfied except where some subterm falls into the 'hole'. We looked for incomplete models as well as for complete ones.

The full list of previously open cases of QG1 QG7 now closed by one or other of the three programs is as follows.

QG2 Incomplete model of QG8.2 (order 8 with missing sub-square of order 2) does not exist.

QG3 Idempotent models of order 12 exist.

QG4 Idempotent models of order 12 exist. Incomplete idempotent models of orders 10.2 and 11.2 exist.

QG5 No model of order 10 exists. No idempotent model oforder 12, 13 or 14 exists. No incomplete idempotent model of order v.k exists for $1 < k < v < 12$.

QG6 Model of order 9 exists. No model of order 7, 10, 11 or 12 exists. No incomplete idempotent model of order v.k exists for $1 < A: < v < 12$.

QG7 No model of order 7, 8, 10, 11 or 12 exists.

One additional negative result is that there is no self-orthogonal idempotent quasigroup of order 12 satisfying the equation

$$a(ba) = b$$

It is known that there are such quasigroups of every order congruent to 1 (mod 3) except order 10, and that there is no such quasigroup of order congruent to 2 (mod 3). It is not known whether there is one of order congruent to 0 (mod 3).

References

[Bennett, 1989] F. E. Bennett, Quasigroup Identities and Mendelsohn Designs, Canadian Journal of Mathematics 41 (1989), pp. 341 368.

[Bennett and Zhu, 1992] F. E. Bennett & L. Zhu, Conjugate-Orthogonal Latin Squares and Related Structures, Contemporary Design Theory: A Collection of Surveys, ed J. H. Dinitz & D. R. Stinson. New York, Wiley, 1992.

[Denes and Keedwell, 1974] J. Denes & A. D. Keedwell, Latin Squares and their Applications, Academic Press, New York, 1974.

[Fujita et al, 1992] M. Fujita, R. Hasegawa, M. Koshimura, & H. Fujita, Model Generation Theorem Provers on A Parallel Inference Machine, Proc. of FGCS'92, Tokyo, 1992.

[Fuchi, 1990] K. Fuchi, Impression on KL1 programming—from my experience with writing parallel provers, Proc. of KL1 Programming Workshop '90, pp.131-139, 1990 (in Japanese).

[Fujita and Hasegawa, 1991] H. Fujita and R. Hasegawa, A Model Generation Theorem Prover in KL1 Using Ramified-Stack Algorithm, Proc. of ICLP91, pp.535-548, 1991.

[Lam et al, 1989] C. W. H. Lam, L. Thiel & S. Swiercz, The Non-existence of Finite Projective Planes of Order 10, Canadian Journal of Mathematics 41 (1989), pp. 1117-1123.

[Manthy and Bry, 1988] R. Manthey, & F. Bry, SATCHMO: a theorem prover implemented in Prolog, Proc. of CADE-9, Springer-Verlag, 1988.

[McCune, 1990] W. W. McCune, OTTER 2.0, Proc. of CADE-10, Springer-Verlag, 1990, pp. 663-664.

[Slaney, 1992] J. K. Slaney, FINDER, Finite Domain Enumerator: Notes and Guide, technical report TR-ARP-1/92, Automated Reasoning Program, Australian National University, 1992.

[Stickel, 1988] M. E. Stickel, A Prolog Technology Theorem, Prover: Implementation by an Extended Prolog Compiler, Journal of Automated Reasoning 4 (1988), pp. 353 380.

[Wos et al, 1990] L. Wos, S. Winker, W. McCune, R, Overbeek, E. Lusk, R. Stevens & R. Butler, Automated Reasoning Contributes to Mathematics and Logic, Proc. of CADE-10, Springer-Verlag, 1990.

[Zhang, 1991] J. Zhang, Private communication, 1991.