More on quasigroup problems* (Extended Abstract)

Masayuki Fujita
Mitsubishi Research Institute, Inc.
Tokyo, Japan
{mfujita@mri.co.jp

December, 1994

1 Introduction

An MGTP(model generation theorem prover)[5] on the Parallel Inference Machine with 256 processors, [4] made an obvious breakthrough in deciding some finite quasigroup existence problems by a method of model enumeration, after some tries and a success in China[10] and Australia[7].

Since then, two types of systems, such Mark Stickle's DDPP, an implementation of the Davis-Putnam algorithm, and John Slaney's Finder, a finite-domain constraint based enumeration system, have achieved solutions to harder problems in the domain and contributed to great progress in design theory in discrete mathematics[8]. In this process, the major reasons for new breakthroughs are strongly related to "how to minimize guessing," or how to cleverly choose the next place to assign a value. Choosing a cell with the least alternative values in the latin square or choosing a literal in one of the least length positive clauses is the common criterion of all systems. And heuristics we found have taken the role of shortening the least length positive clauses. A list of the heuristics and the reasons why they are effective will clarify the advantage of representing the problem as propositions. One reason is that all heuristics are represented declaratively and they are independent from the program and are very easy to verify and modify.

Further investigation was made since then and the following two parts of this talk are an interim report of the continuing research.

1. Experimental results by binary clause clusure method[3]
2. Further challenging open problems

2 Experiments

In the research domain of finite constraint solving, look ahead is a well known method to propagate constraints[9]. This method is obviously useful for model finding. This has quite improved the search with the high cost of execution time in the quasigroup application [6]. Because look ahead compute again and again all possible unit consequences of each undecided assignment at every node of the search tree. This is exponentially redundant.

The deductive closure of binary clauses(2-Closure) is not so expensive as look ahead and useful to obtain a similar constraint propagation. Propagation of the constraints by 2-Closure occurs when $a \vee b$ and $a \vee \neg b$ are in the closure.

*This research is supported by the ICOT

Problem	Order	BCCP(time:sec)	Davis Putnam(time:sec)	Look Ahead
QG3	8	236(101)	1037(76)	-
QG5	10	27(75)	38(66)	5
QG5	11	43(225)	136(228)	-
QG5y	12	191(950)	443(883)	83

Table 1: Failed branches in the search space

But the look ahead is not equivalent to this. For example, there is a well known method in the constraint domain that the look ahead simulates. In this method, 'supported' or 'non-supported' assignment corresponds to positive or negative fact respectively. A supported assignment $X_i = a$ by X_j is that all possible assignment of X_j derives $X_i = a$. So all the other possible assignment of Xi are negated by this fact. Non-supported assignment is one of the dual constraint propagation of this situation.

For seeing how the above method work to the quasigroup problems, an experimental program BCCP(binary clause closure program) in klic was created and some results1 were obtained. The results of DDPP and Look Ahead are from [8] and [6].

We can see the look ahead has much smaller size of the search tree than BCCP. However BCCP runs reasonably fast and using a basic data structure instead of stack vectors can make some constant times speedup.

3 More challenging problems

The second part is the introduction of further open problems in this domain. Problems are from the reffered pages of the book [2]. Although the order is comparable to QG1 - QG8 in FSS93, none of them are solved by our method. QG11 was stated 'out of the existing computer's power' in the book.

QG9 (p14)

for order n quasigroups, for max size t of partial transversial $t \leq n-1$. For odd n, $t = n$. No counterexamples are found for any order
(partial transversial: a set of cells of a quasigroup with distinct value and all of them are distinct both in row and column)

QG10 (p15)

Are there three mutually orthogonal LS(10) ?

QG11 (P50)

Are there order 11 row complete lattin square
(complete:all $n(n-1)$ ordered pairs$(A_{i,j}, A_{i,j+1})$ are distinct)

QG12 (P116)

Construct N_∞-LS(16)
(N_∞: for all LS(N) for all $1 < I < N$ no subsquare exists)

QG13 (p117)

are there N_∞ -LS(N) other than $N = 2\ a * 3\ b$?

QG14 (P145)

ISOLS(14,4)(Idempotent Self Orthogonal Latin Square order 14 with a hole of size 4) ?

References

[1] F. Bennett, *Quasigroup Identities and Mendelshon Designs*, **Canadian Journal of Mathematics** 41 (1989), pp. 341–368.

[2] J. Dénes, A.D. Keedwell, ed., *Latin Squares: New Developments in the Theory and Application*, North-Holland, 1991.

[3] A. Gelder and Y. Tsuji, *Satisfiability Testing with More Reasoning and Less Guessing* Preprint, 1994.

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

[5] M. Fujita, J. Slaney & F. Bennett, *Automatic Generation of Some Results in Finite Algebra*, **Proc. International Joint Conference on Artificial Intelligence**, 1993.

[6] Masayuki Fujita, Frank O'Carroll, *Heuristics and more Heuristics:Toward Solving Harder Quasigroup Problems (Extended Abstract)*, **workshop on Automated Reasoning In Algebra, CADE-12, 1994**

[7] J. Slaney, *FINDER, Finite Domain Enumerator: Version 2.0 Notes and Guide*, technical report TR-ARP-1/92, Automated Reasoning Project, Australian National University, 1992.

[8] J., Slaney, M. Fujita, & M. Stickel., *Automated reasoning and exhaustive search: Quasigroup existence problems*,**Computers and Mathematics with Applications**, 1993.

[9] A.K.M., Tan, *Seach Strategies in Finite Constraint Satisfaction*, to appear, 1994 Honours Thesis, Australian National University, 1994.

[10] J. Zhang, *Search for Idempotent Models of Quasigroup Identities*, Typescript, Institute of Software, Academia Sinica, Beijing.