Channeling Constraints and Value Ordering in the QuasiGroup Completion Problem

Ivan Dotii and Alvaro del Val and Manuel Cebrian
Departamento de Ingenieria Informatica
Universidad Autonoma de Madrid

1 Introduction

The Quasigroup Completion Problem (QCP) is a very challenging benchmark among combinatorial problems, and a focus of much recent interest in the area of constraint programming. It has numerous practical applications [Gomes *et al,* 2002]; it has been put forward as a benchmark which can bridge the gap between purely random instances and highly structured problems [Gomes *et al,* 2002]; and its structure as a multiple permutation problem [Walsh, 2001] is common to many other important problems in constraint satisfaction.

I Gomes *et al,* 2002] reports that QCPs of order 40 could not be solved by pure constraint programming approaches, but could sometimes be solved by hybrid approaches combining constraint programming with mixed integer programming techniques from operations research. In this paper, we show that the pure constraint satisfaction approach can solve many problems of order 45 in the transition phase, which corresponds to the peak of difficulty. Our solution combines a number of known ideas -the use of redundant modeling [Cheng *et al,* 1996] with primal and dual models of the problem connected by channeling constraints [Walsh, 2001]- with some novel aspects, as well as a new and very effective value ordering heuristic.

2 Models for QCPs

A quasigroup is an ordered pair (Q,), where *Q* is a set and •is a binary operation on *Q* such that the equations a • *x* = *b* and *y* ■ *a* = *b* are uniquely solvable for every pair of elements a, *b* in *Q* [Gomes *et al,* 2002]. The order n of the quasigroup is the cardinality of the set *Q.* A quasigroup can be seen as an n x *n* multiplication table defining a Latin Square, which must be filled with "colors" so that the colors of each row are all distinct, and similarly for columns. A QCP is the problem of coloring a partially filled Latin square, which is known to be NP-Complete [Colbourn, 1984].

Each row and column of a Latin Square defines a permutation problem, i.e. a constraint satisfaction problem with the same number of variables as values, where a solution is a permutation of the values [Walsh, 2001]. The QCP is a multiple permutation problem with 2n intersecting permutation constraints (n row permutation constraints and n column permutation constraints). In order to represent those permutation problems we have implemented three models:

- *Primal Model:* This model has a variable $x_{i,j}$ for each cell of the latin square, with *i* and *j* its coordinates. Their possible values are the colors of the cell.
- *Row Dual Model:* This is the first dual model, it represents the colors in each row; the dual variable $r_{i,k}$ is the k-th color in the i-th row, and the values it can take are the columns where it can be placed.
- *Column Dual Model:* This one models the colors in each column; the dual variable $c_{j,k}$ represents the k-th color in the j-th column, with rows as its possible values.

The primal and dual not-equal constraints specify that the colors in a row or column must all be different. While these are sufficient to capture the problem (and in fact redundantly so), it is known that these models can be improved by connecting them through *channeling constraints:*

- *Row Channeling Constraints:* Constraints for the n row permutation constraints, linking the primal model with the row dual model: $x_{i,j} = k \Leftrightarrow r_{i,k} = j$.
- *Column Channeling Constraints:* Corresponding to the n column permutation constraints, they link the primal and dual column models: $x_{i,j} = k \Leftrightarrow c_{j,k} = i$.
- *Triangular Channeling Constraints:* These (novel) constraints link both dual models, closing a "triangle" among the three models: $c_{j,k} = i \Leftrightarrow r_{i,k} = j$.

It's easy to show that the first two kinds of channeling constraint fully capture a QCP (see also [Walsh, 2001]), making the primal and dual constraints redundant.

3 Value Ordering

Key to our results is a new *value ordering* heuristic, which we might call the *min-domain value selection heuristic.* Once a primal or dual variable is selected, we need to choose a value for it. Since any such value corresponds to one specific variable from each of the two other models, we select the value whose corresponding two variables have a minimal "combined" domain. Specifically, say we have chosen *X{j.* Then we choose a color *k* from its currently active domain for which the sum of the current domain sizes of $r_{i,k}$ and $C_{J,K}$ is minimal among the currently available colors for $x_{i,j}$.

4 Results

We focused our experiments on solvable instances. To make our results comparable with those reported in [Gomes *et al,*

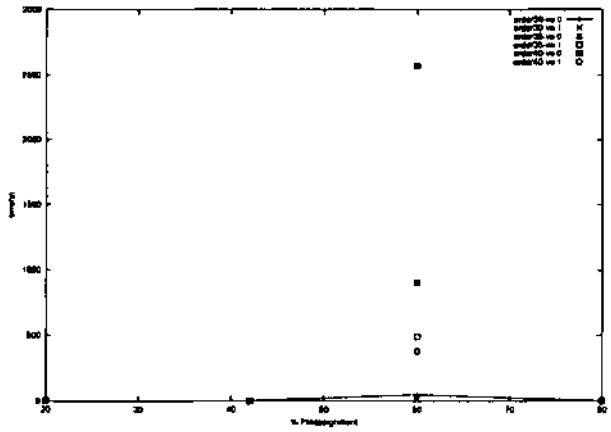

Figure 1: Mean solution time on QCPs of order 30, 35 and 40 with (vol) and without value ordering (voO).

2002], we used the same generator[1], and considered only "balanced" instances, which are known to be the hardest. We used an (AC3-based) Generalized Arc Consistency algorithm based on van Beek's CPLAN implementation (http:ai.uwaterloo.ca/~vanbeek/software/software.html), with a few improvements. We considered various (complete) combinations of all the above mentioned constraints, and found that adding the primal and dual "not-equal" constraints never helped performance in the presence of row and column channeling constraints, and yielded even worse results in their absence. Most of the results discussed below therefore use a model with only row and column channeling constraints; they also use "MAC" (mantaining arc consistency [Bessiere, Regin, 19961). Since our preliminary tests showed that CBJ and nogood learning seem to never help in these problems, and are often a source of significant overhead, we discarded them. We also considered various variable selection heuristics. We found the *min-domain* heuristic, which selects a variable with smallest domain to be uniformly the best among the alternatives we tried, which included more sophisticated heuristics such as *dom+degree, dom/degree,* and variants of the above which took into account the (primal or dual) model to which variables belong, e.g. selecting only among primal variables, etc.

With only row and column channeling constraints, our implementation outperforms classical approaches, in that we can solve a considerable amount of instances of order 40 in a reasonable time. As mentioned above, this was very recently considered as out of reach for pure constraint programming approaches [Gomes *et al,* 2002], which ours is (in fact, we don't even use global or non-binary constraints). The results when this model was used with the min-domain value ordering heuristic were quite surprising, as it outperformed previous tests in three orders of magnitude in some cases. For example, for the instance bqwh-35-405-5.pls (balanced instance of order 35 and 60% preassigned) it took 2905 sees without value ordering and only 0.40 sees with it. In Figure 1 we present a comparison of our results with and without value ordering for quasigroups of order 30,35 and 40, and 60% preassignment, close to the phase transition [Gomes *et al,* 2002].

Encouraged by this performance, we generated some instances of order 45 and 60% preassignment. Again the value ordering heuristic turns out to be so efficient that we were able to solve a great number of instances in a reasonable amount of time. The next table shows median and mean time in seconds (the latter taken only over solved instances) and percent of solved instances, using value ordering for problems in the transition phase.

[order	mean	median	% solved	timeout
30	148.84	174.11	68%	1000
35	533.43	163.48	84%	3600
40	690.85	842.11	68%	5000
45	1090.10	2971.40	56%	6000

Finally, we are currently exploring the effect of adding "triangular" channeling constraints; our preliminary results suggest that they can improve performance by an additional 30-35%, using in this case only forward checking (e.g. from 902 to 666 seconds in one order 45/60% preassigned instance). The idea is that they provably allow forward checking to perform as much pruning as arc-consistency, while saving many useless checks; in fact, as many as 60% fewer checks, even though the savings in time are not as great due to effects we are currently exploring.

5 Conclusions and Future Work

We have shown that a pure CSP approach can handle quasigroup completion problems significantly larger than was thought possible, using appropriate models and value ordering heuristics, and even in the absence of global alldiff constraints. Yet our results are preliminary, and no doubt it would be interesting to consider other value ordering heuristics and ways to make the triangular channeling constraints most effective.

References

[Gomes *et al,* 2002] C. Gomes and D. B. Shmoys. The promise of LP to Boost CSP Techniques for Combinatorial Problems. In *Proc. of CPAIOR'02,* Le Croisic, March 2002.

[Gomes *et al,* 2002] C. Gomes and D. B. Shmoys. Completing Quasigroups or Latin Squares: A Structured Graph Coloring Problem In *Proc. Computational Symposium on Graph Coloring and Generalizations,* 2002.

[Walsh, 2001] T. Walsh. Permutation Problems and Channeling Constraints. TR 26, APES Group, 2001.

[Colbourn, 1984] C. Colbourn. The complexity of completing partial latin squares. In *Discrete applied Mathematics,* (8):25-30, 1984.

[Cheng *et al,* 1996] B.M.W. Cheng, J.H.M. Lee, and J.C.K Wu. Speeding up constraint propagation by redundant modeling. In *2nd Int. Conf. on Principles and Practice of Constraint Programming,* pp. 91-103, 1996.

[Bessiere, Regin, 1996] C.Bessiere and J.C.Regin Mac and combined heuristics: two reason to forsake FC (and CBJ?) on hard problemss. In *CP'96,* pp. 61-75, Cambridge MA.

'Generator *Isencode,* kindly provided by Carta Gomes.