A Compact and Efficient SAT-Encoding of Finite Domain CSP

Tomoya Tanjo[1], Naoyuki Tamura[2], and Mutsunori Banbara[2]

[1] Graduate School of Engineering, Kobe University, Japan
[2] Information Science and Technology Center, Kobe University, Japan
tanjo@stu.kobe-u.ac.jp, tamura@kobe-u.ac.jp, banbara@kobe-u.ac.jp

Extended Abstract

A (finite) Constraint Satisfaction Problem (CSP) is a combinatorial problem to find an assignment which satisfies all given constraints over finite domains. A SAT-based CSP solver is a program which solves a CSP by encoding it to SAT and searching solutions by SAT solvers. Remarkable improvements in the efficiency of SAT solvers make SAT-based CSP solvers applicable for solving hard and practical problems. A number of SAT encoding methods have been therefore proposed: direct encoding, support encoding, log encoding, log-support encoding, and order encoding.

Among them, *order encoding* [4] has showed a good performance for a wide variety of problems, including Open-Shop Scheduling problems, two-dimensional strip packing problems, and test case generation. Its effectiveness has also been shown by the fact that a SAT-based CSP solver Sugar [1] became a winner in several categories of the 2008 and 2009 International CSP Solver Competitions.

However, in the order encoding, the size of SAT-encoded instances becomes huge when the domain size of the original CSP is large. On the other hand, the *log encoding* [3,1] uses a bit-wise representation for integer variables. The size of SAT-encoded instances is therefore compact (linear to $\log d$), but its performance is slow in general because it requires many inference steps to "ripple" carries.

In this paper, we propose a new encoding, named *compact order encoding*, aiming to be compact and efficient. The basic idea of the compact order encoding is the use of a numeric system of base $B \geq 2$. That is, each integer variable x is represented by a summation $\sum_{i=0}^{m-1} B^i x_i$ where $m = \lceil \log_B d \rceil$ and $0 \leq x_i < B$ for all x_i, and each x_i is encoded by the order encoding.

Each ternary constraints of addition and multiplication can be encoded into at most $O(B^2 \log_B d)$ and $O(B^3 \log_B d + B^2 \log_B^2 d)$ clauses respectively which are much less than $O(d^2)$ clauses of the order encoding. The compact order encoding can generate much efficient SAT instance than the log encoding in general because it requires fewer carry propagations. Please note that the compact order encoding with base $B = 2$ is equivalent to the log encoding, and the one with base $B \geq d$ is equivalent to the order encoding.

[1] http://bach.istc.kobe-u.ac.jp/sugar/

K.A. Sakallah and L. Simon (Eds.): SAT 2011, LNCS 6695, pp. 375–376, 2011.

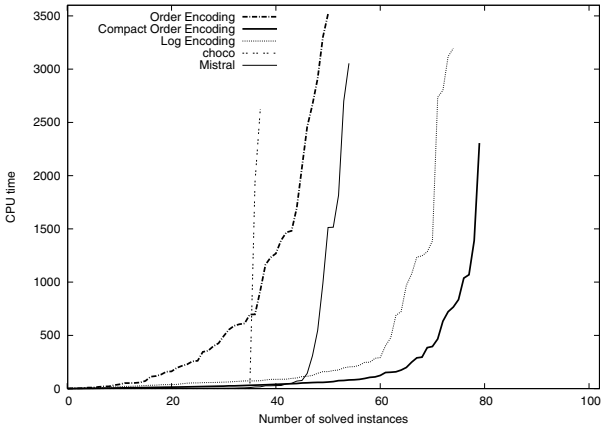

Fig. 1. Cactus plot of different encodings, choco, and Mistral for OSS instances

To evaluate the effectiveness and scalability of our encoding, we used the most difficult series of Open-Shop Scheduling (OSS) benchmark set by Brucker *et al.* We also used the instances with very large domain sizes, which are generated from OSS instances by multiplying the process times by constant factor $s \in \{1, 10, 20, 100, 200, 1000\}$. The performance of the compact order encoding with $m = 2$ (i.e. $B = \lceil d^{\frac{1}{2}} \rceil$) is compared with those of the order and log encodings in addition to the state-of-the-art CSP solvers choco 2.11 [5] and Mistral 1.550 [2].

Fig. 1 shows the cactus plot of benchmark results in which the number of solved instances is on the x-axis and the CPU time is on the y-axis. The compact order encoding solved the most instances for almost any CPU time limit and it solved large instances which could not be solved by order solvers.

As future work, we plan to investigate the choice of appropriate base B for solving a wide variety of problems.

References

1. Gelder, A.V.: Another look at graph coloring via propositional satisfiability. Discrete Applied Mathematics 156(2), 230–243 (2008)
2. Hebrard, E.: Mistral, a constraint satisfaction library. In: Proceedings of the 3rd International CSP Solver Competition. pp. 31–39 (2008)
3. Iwama, K., Miyazaki, S.: SAT-variable complexity of hard combinatorial problems. In: Proceedings of the IFIP 13th World Computer Congress, pp. 253–258 (1994)
4. Tamura, N., Taga, A., Kitagawa, S., Banbara, M.: Compiling finite linear CSP into SAT. Constraints 14(2), 254–272 (2009)
5. The choco team: choco: an open source Java constraint programming library. In: Proceedings of the 3rd International CSP Solver Competition, pp. 7–13 (2008)