MDPI

Communication

Fractional Encoding of At-Most-K Constraints on SAT

Miki Yonekura [1,†], Shunji Nishimura [2]

[1] Affiliation 1; mikiyonekura4040@gmail.com
[2] Affiliation 2; s-nishimura@@oita-ct.ac.jp
* Correspondence: mikiyonekura4040@gmail.com; Tel.: +81-70-2392-7093
† Current address: 1666 Maki, Oita City, Oita Prefecture, Japan

Abstract: The SAT problem, a classical NP-complete problem, poses computational complexity challenges. Nevertheless, progress in algorithms and hardware has allowed the resolution of logic formulas with up to 1 million variables and 10 million constraints, depending on the problem. As a result, converting real-world problems to SAT and utilizing SAT solvers is prevalent in various fields like model checking and electric circuit generation. The bottleneck in the computation time of solving SAT is often the At-Most-K constraints, which are satisfied only if at most K of the whole logic variables are true. The Pairwise Encoding method, the simplest encoding method for At-Most-K constraints, requires $O(n^2)$ size of logical expressions for n target variables, resulting in an exponential increase in logical expression size. This study introduces the Fractional Encoding method as a means to decrease the size of logical expressions for At-Most-K constraints. Fractional Encoding splits the set of variables into m parts and applies Pairwise Encoding to each subset, reducing the number of target variables to $1/m$. However, simply splitting the set results in a significant decrease in the number of possible combinations of variables that can be the solution. Therefore, to avoid that problem, Fractional Encoding uses auxiliary variables to increase or decrease the number of true variables among split sets. Comparison with conventional methods shows that the size of the logic expression can be reduced when the number of target variables increases.

Keywords: At-Most-K Constraints; Satisfiability Problem; Logical Expression

1. Introduction

The satisfiability problem (SAT) is a classical NP-complete problem and is difficult from the viewpoint of computational complexity. However, recent advances in algorithms and hardware have made it possible to solve even large-scale real-world problems at high speed. Therefore, the approach of converting (encoding) the real-world problem to SAT and solving it is used in various fields such as software model checking and electric circuit generation [1].

The real-world problem encoded in SAT consists of various constraints, but the At-Most-K constraints often become a bottleneck of computational complexity [2]. Therefore, in this study, we propose Fractional Encoding as an encoding method for At-Most-K constraints to suppress the increase in computational complexity.

Satisfiability Problem(SAT)

"Solving SAT" means determining the satisfiability of a propositional logic formula. In other words, it determines whether there exists an assignment (model) of propositional variables to the logical formula containing propositional variables such that the formula becomes true. A program that solves SAT is called a SAT solver, and it determines whether a given SAT is satisfiable or unsatisfiable. There are various types of SAT solvers, such as MiniSat [3], which is well-known, and CaDiCaL [4], which has achieved excellent results in recent competitions. In most SAT solvers, if the SAT is satisfiable, a concrete assignment is shown; if it is unsatisfiable, the assignment is shown to be non-existent.

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published:

The logical expressions encoded in SAT are in conjunction standard form (CNF). CNF is a form of expressing logical expressions by a sequence of disjunction clauses and is the target of a SAT solver [5].

At-Most-K Constraints

SAT is composed of various constraints (logical expressions). The At-Most-K constraint is the constraint that no more than K variables can be true. As the simplest example, the following is a Pairwise Encoding with at most one true variable and three target variables.

$$(\neg x_1 \vee \neg x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_3) \tag{1}$$

Although Pairwise Encoding is a simple implementation, it requires $O(n^2)$ clauses for n target variables, which becomes a huge expression when n is large. Therefore, various coding methods have been proposed to suppress the number of clauses by introducing auxiliary variables. Typical coding methods that use auxiliary variables include Binary Encoding and Counter Encoding[6,7,8].

Binary encoding introduces the auxiliary variables B1,...., Blog2n, and by mapping each target variable Xi to a unique bit sequence Si.

The encoding method proposed in this study, "fractional encoding," also suppress the number of clauses better than pairwise encoding by introducing auxiliary variables.

2. Methods

In this study, we propose an encoding method with At-Most-K constraints that distributes the computational complexity by splitting the set of target variables into multiple parts. The proposed method is called Fractional Encoding because it is based on the concept that K is the numerator and the number of target variables, n, is the denominator. Hereafter, when there are at most K variables that are true out of n target variables, it is denoted as $AtMost\ K/n(a\ set\ of\ target\ variables)$.

Splitting target variables

By simply splitting the set of target variables into m subsets and applying Pairwise Encoding to each subset, the number of target variables n can be reduced to $1/m$. In the following example, a set of 8 target variables is split into two subsets.

$$AtMost\ 4/8(x_1,...,x_8) \xrightarrow{Split\ in\ two} AtMost\ 2/4(x_1,...,x_4) + AtMost\ 2/4(x_5,...,x_8) \tag{2}$$

The number of clauses in Pairwise Encoding is calculated by ${}_nC_{K+1}$ and can be reduced from 56 to 8 as shown in (3) below. However, this example does not lead to $assignment = x_1, x_2, x_3, x_5$ such that 3 variables in set $x_1,...,x_4$ and 1 variable in set $x_5,...,x_8$ are true. Thus, a simple splitting of the set greatly reduces the number of possible combinations of variables that can be solved (solution space).

$$\underbrace{{}_8C_5}_{\text{56 clauses}} \xrightarrow{Split\ in\ two} \underbrace{{}_4C_3 + {}_4C_3}_{\text{8 clauses}} \tag{3}$$

Fractional Encoding

Fractional Encoding realizes At-Most-K constraints by propagating the fraction from the upper layer to the lower layer, as in the tree structure shown in Fig. 1 Propagation is performed using auxiliary variables($g_{i,j}$) to dynamically determine the At-Most-K constraints to be applied to the split target variables. This prevents the solution space from

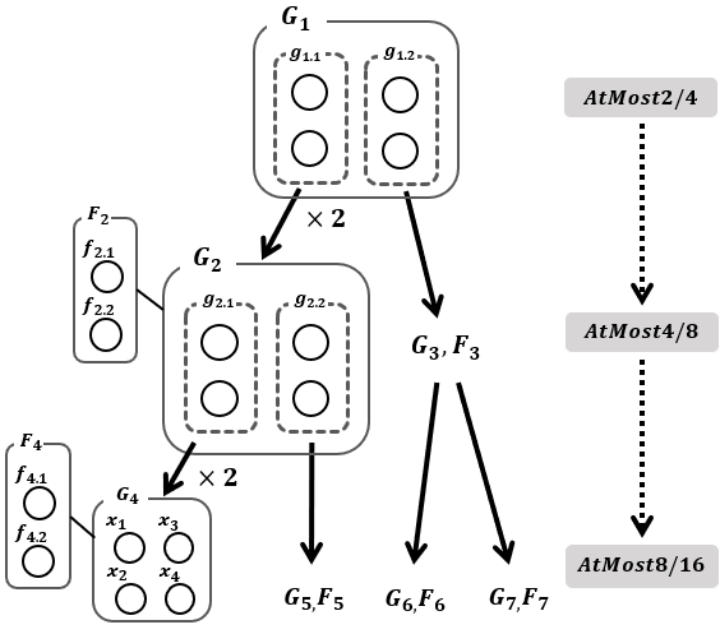

Figure 1. This figure shows the tree structure for generating $AtMost\ 8/16(x_1, ..., x_{16})$ using Fractional Encoding. The bottom layer serves as the target variables, and the other layers play the role of auxiliary variables. Each layer has a group G_i containing four variables, the number of which depends on the number of layers. The number of layers is $\log_2 K$, and the number of groups G_i is calculated as $1/2K \log_2 K$. In addition, each group is assigned a fine-tuning variable F_i(to be explained below) to determine its state.

decreasing. As an example, the splitting of target variables for the $AtMost\ 8/16(x_1, ..., x_{16})$ constraint is shown below.

$$\begin{aligned} &AtMost\ 8/16(x_1, ..., x_{16}) \\ &\xrightarrow{Split\ in\ four} AtMost\ k_1/4(x_1, ..., x_4) \\ &\quad + AtMost\ k_2/4(x_5, ..., x_8) \\ &\quad + AtMost\ k_3/4(x_9, ..., x_{12}) \\ &\quad + AtMost\ k_4/4(x_{13}, ..., x_{16}) \end{aligned} \tag{4}$$

In (4), the 16 target variables are split into four subsets of $G_4 = \{x_1, ...x_4\}, ..., G_7$. $k_1, ..., k_4$ are dynamically determined values whose total value is less than or equal to K, as shown below.

$$k_1 + k_2 + k_3 + k_4 = K \tag{5}$$

The variables belonging to $G_1, ..., G_3$ become auxiliary variables that propagate the ratio of the number of variables that can be true to the number of target variables as shown on the right in Figure 1 to realize (4) and (5). In addition, since the number of variables that become true at the bottom layer is controlled by propagating the ratio set at the top layer to the bottom layer, there is a fraction(Top layer K/n) that serves as the base. When 2/4 is used as the base, only $2 \times 2^m/4 \times 2^m$ patterns can be generated, such as At-Most-2/4, At-Most-4/8, and At-Most-8/16. The encoding procedure based on 2/4 is shown below.

1. **Ratio setting at the top layer** In the formula below, the At-Most-2/4 constraint is applied to the top layer to set the base 2/4 ratio (at most half of the target variables are true).

$$AtMost2(G_1) \tag{6}$$

2. **Introduction of fine-tuning variables** The fine-tuning variable $F_i = \{f_{i.1}, f_{i.2}\}$ is a variable to increase or decrease the number of variables that can be true among groups G_i of the same layer. Each group has three states $plus, minus, const$ and is uniquely encoded using two fine-tuning variables. Depending on the state of each set, the variables that can be true are increased or decreased as shown in Figure 2.

$$plus(i) : f_{i.1} \wedge f_{i.2} \tag{7}$$
$$minus(i) : \neg f_{i.1} \wedge \neg f_{i.2} \tag{8}$$
$$const(i) : f_{i.1} \oplus f_{i.2} \tag{9}$$

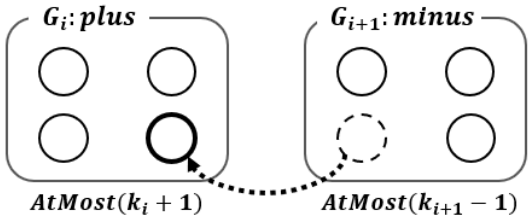

Figure 2. As shown in this figure, If the state is $plus$, the number of variables that can be true in the group is $+1$, and if $minus$, it is -1.

3. **Added logical expressions to propagate ratio to lower layer** The following equation adds a constraint that propagates the ratio from the upper to the lower layer. In addition, $p(p = 1/2k \log_2 k - 1)$ indicates the number of Local-Propagations, which will be discussed later. The Exactly-K constraint(to be explained below) in the upper layer counts the auxiliary variables that become true, and the At-Most-2K+z constraint (z=-1,0,1) is added in the lower layer.

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} Exactly0(g_{i.j}) \wedge const(2i + j) \Rightarrow AtMost0(G_{2i+j}) \tag{10}$$

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} (Exactly0(g_{i.j}) \wedge plus(2i + j)) \vee (Exactly1(g_{i.j}) \wedge minus(2i + j)) \Rightarrow AtMost1(G_{2i+j}) \tag{11}$$

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} Exactly1(g_{i.j}) \wedge const(2i + j) \Rightarrow AtMost2(G_{2i+j}) \tag{12}$$

$$\bigwedge_{i=1}^{p}\bigwedge_{j=0}^{1}(Exactly1(g_{i.j}) \wedge plus(2i+j)) \vee (Exactly2(g_{i.j}) \wedge minus(2i+j)) \Rightarrow AtMost3(G_{2i+j}) \tag{13}$$

The Exactly-K constraint is a constraint that counts exactly K variables to be true. It is expressed by the At-Most-K constraint and the At-Least-K constraint that indicates that there are at least K variables that are true, as shown below.

$$ExactrlyK \Leftrightarrow AtMostK \wedge AtLeastK \tag{14}$$

The leaf variables (target variables) generated by (10), (11), (12), and (13) propagate the proportions set at the top layer. Also, when Exactly-0($g_{i.j}$), the number of possible true values in group G_{2i+j} cannot be further reduced, and when Exactly-2($g_{i.j}$), the number of possible true values in group G_{2i+j} cannot be further increased, so we add the following equation.

$$Exactly0(g_{i.j}) \Rightarrow \neg minus(2i+j) \tag{15}$$
$$Exactly2(g_{i.j}) \Rightarrow \neg plus(2i+j) \tag{16}$$

4. **Add constraints for plus/minus offsetting at each layer** By applying At-Most-2^m to the fine-tuning variables in each layer, the pluses and minuses in the same layer must be balanced out. This allows increasing or decreasing the number of variables that can be true among the groups G in the same layer. In the equation below, p represents the number of layers and is calculated as in $q = number\ of\ layers$.

$$\bigwedge_{m=1}^{q} AtMost2^m(f_{2^m.1}...f_{2^{m+1}-1.2}) \tag{17}$$

Pattern Extension by Fixing Variables

Since Fractional Encoding generates At-Most-K constraints based on the base fraction, it is difficult to handle an arbitrary number of target variables. For example, if you want to create At-Most-3/5, there is no suitable fraction(top layer K/n). Therefore, we extend the patterns that can be generated by fixing values of a part of the target variables. Fixing true/false means adding clauses that make certain target variables true or false, as shown below.

$$Fix\ x_i\ as\ true : x_i \tag{18}$$
$$Fix\ x_i\ as\ false : \neg x_i \tag{19}$$

As shown below, when number c of target variables are fixed as true, K and n decrease by c, and when they are fixed as false, only n decrease by c.Figure 3 shows an example of three variables fixed.

$$Fix\ c\ true :\ AtMost\frac{K-c}{n-c} \tag{20}$$
$$Fix\ c\ false :\ AtMost\frac{K}{n-c} \tag{21}$$

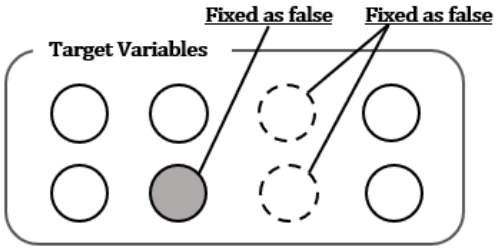

Figure 3. In the example, by fixing three variables, we transformed At-Most-4/8 to At-Most-3/5. By extending the possible patterns that can be generated in this way, more real-world problems can be solved.

Analysis

We discuss the number of clauses of the At-Most-K constraint generated by Fractional Encoding. Hereafter, the propagation from $g_{i.j}$ to G_{2i+j}, which is important in the clauses number calculation, will be called Local-Propagation.

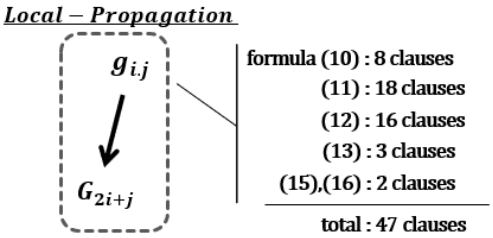

Figure 4. The number of clauses required when each equation required for Local-propagation is converted to CNF. The total of these is the number of clauses required per Local-Propagation, which is 47.

The number of clauses of At-Most-K/n constraints encoded by Fractional Encoding can be transformed as follows.

$$\begin{aligned} &clauses(AtMostK/n) = \\ &\quad clauses(AtMost\frac{K}{2^1}/\frac{n}{2^1}) + clauses(AtMost\frac{K}{2^2}/\frac{n}{2^2}) \\ &\quad ... + clauses(AtMost2/4) + clauses(Local\text{-}Propagations) \end{aligned} \tag{22}$$

By recursively transforming, the number of clauses in the At-Most-K/n constraint can be expressed as the number of Local-Propagations required to generate the At-Most-K/n constraint × 47. The number of Local-Propagations depends on the number of layers ($r = log_2 K$) and is calculated to be $O(K \log_2 K)$ as follows. The number of Local-Propagations depends on the number of layers; the first term is clauses(Local-Propagation) in (22); the second term shows the clauses(Local-Propagation) that are added when (22) is Calculated recursively.

$$\begin{aligned} (2^r - 2) + \sum_{x=0}^{r-3} 2^x \times ((2^{r-1-x} - 2)) \\ = r \times 2^{r-1} - 2^{r-1} \\ = \frac{1}{2} K(\log_2 K - 1) \end{aligned} \tag{23}$$

$$clauses : 47 \times \frac{1}{2} K(\log_2 K - 1) \tag{24}$$

There are two types of auxiliary variables that Fractional Encoding requires: the first is the variables that propagate the ratio to the bottom variable (the target variable), which contains four auxiliary variables in each group G. The second is the fine-tuning variables for each group G, two for each group G The second is a fine-tuning variable for each group G, two for each group G. The calculation of the number of auxiliary variables is shown below: 6(2+4) auxiliary variables are needed for each group G, as shown in the first term. In addition, since the fine-tuning variables are not needed for the top-layer group G_T, they are subtracted as shown in the second term. In addition, variables in the lowest group G_B are target variables and are subtracted in the third term.

$$6G - 2G_T - 4G_B \tag{25}$$

Since G, G_T, and G_b can be computed with $1/2K\log_2 K$, $1/2K$, and $1/4K(\log_2 K + 1)$, respectively, the number of auxiliary variables becomes D, as shown below.

$$\begin{aligned} &3K\log_2 K - K - (\log_2 K + 1) \\ &\quad = 2K(\log_2 K - 1) \end{aligned} \tag{26}$$

Table 1 shows a comparison with conventional methods regarding the order computational complexity of the number of clauses and auxiliary variables.

Table 1. This table compares the number of clauses and auxiliary variables.

Method	Origin	Clauses	Auxiliary vars
Pairwise	folklore	${}_nC_{k+1}$	0
Binary	Frisch [6,7]	$O(Knlog_2n)$	$O(Kn)$
Counter	Sinz [8]	$O(Kn)$	$O(Kn)$
Fractional	this paper	$O(Klog_2K)$	$O(Klog_2K)$

3. Results

Justification of the proposed method

In order to verify the validity of the proposed method, two verifications were conducted. The first is to verify that the At-Most-K constraint is realized by the proposed method. The At-Most-K constraint is a constraint that the number of variables that can be true is limited to K at most. Therefore, if the proposed method becomes unsatisfiable only when more than K target variables are fixed to be true, it can be shown that the proposed method is correct. To the generated At-Most-K constraints, clauses that fix more than K target variables to be true were added and target to the SAT solver. As a result, the proposed method is correct because it is unsatisfiable only when more than K target variables are fixed as true.

Second, the solution space of the generated At-Most-K constraints was verified: Fractional Encoding has a solution that does not occur when the target variables are split and Pairwise is applied, as described in the "Splitting target variables" section. Therefore,the proposed method can prevent the solution space from decreasing.

Comparison with conventional methods

In order to compare 2/4-based Fractional Encoding with conventional methods, we examined the number of clauses (Fig. 5), the number of auxiliary variables (Fig. 6), and the total number of literals (Fig. 7).

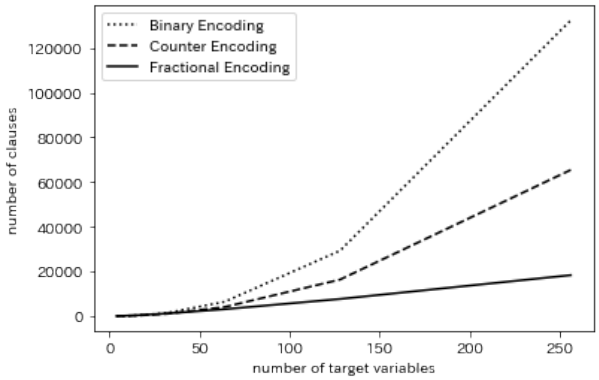

Figure 5. This graph shows the number of clauses in relation to the target variables. The two conventional methods show a significant increase in the number of clauses when the number of target variables exceeds 60. In contrast, Fractional Encoding succeeds in suppressing the number of clauses with a gradual increase.

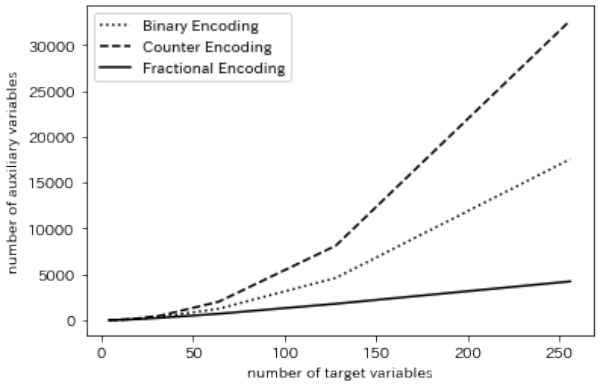

Figure 6. This graph shows the number of auxiliary variables for the target variables. Fractional Encoding also increases gradually and succeeds in suppressing the number of auxiliary variables compared to the conventional methods.

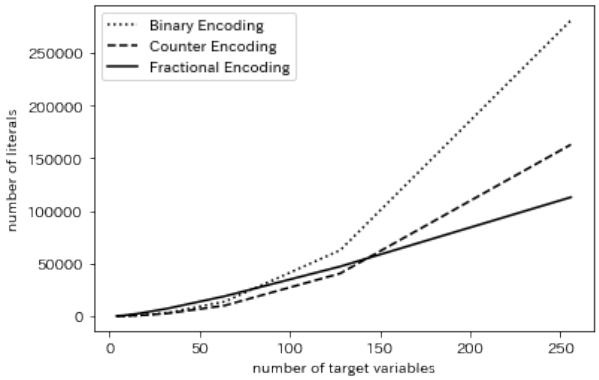

Figure 7. This graph shows the total number of literals for target variables. Fractional Encoding becomes superior to Binary Encoding when the number of target variables exceeds 70, and superior to Counter Encoding when the number of target variables exceeds 120.

4. Discussion

In comparison with the conventional method, it was found that the size of the logical formula can be reduced as the number of target variables increases. This is considered to be because when the number of target variables is small, the number of logical formulas

required for Local-Propagation accounts for a large proportion of the total logical formulas. When the number of target variables is large, the number of logical expressions in Local-Propagation becomes negligible, and the size of logical expressions can be reduced compared to conventional methods. However, Fractional Encoding is inferior to conventional methods in terms of generating flexible At-Most-K constraints. Fractional encoding requires searching for the base fraction (the top layer K/n of the At-Most-K constraints) as described in the section "Fractional Encoding," and if it cannot be found, the modifications described in the section "Pattern Extension with Variable Fixation" must be made. Therefore, it can be said that this method is inferior to the conventional method in terms of generating flexible At-Most-K constraints.

5. Conclusion

In this study, the Fractional Encoding method is proposed as a means of reducing the size of the logical expression of the At-Most-K constraint. Fractional Encoding reduces the size of logical expressions by splitting the set of target variables and using Pairwise Encoding dynamically for each set. However, since simply splitting the set significantly reduces the number of possible variable combinations, we dynamically determined the At-Most-K constraints for the split set using auxiliary variables.

Comparison with conventional methods shows that the size of the logic expression can be reduced when the number of target variables increases. However, when the number of target variables is small or when it is difficult to find the base fraction (Top Layer K/n) necessary for the propagation of the At-Most-K constraints to be generated, the scale of the generated logic formulas is significantly inferior to that of conventional methods.

References

1. Frisch, A.M. SAT Encodings of the At-Most-k Constraint. Some Old, Some New, Some Fast, Some Slow. In*Proc. of the Ninth Int. Workshop of Constraint Modelling and Reformulation* **2010**.
2. Bittner, P.M. SAT Encodings of the At-Most-k Constraint A Case Study on Conguring University Courses. In*Springer Nature Switzerland* **2019**.
3. Een, N. and Sorensson,N. : An extensible SAT-solver. In *Proc. of the 6th International Conference on Theory and Applications of Satisfiability Testing* **2003**.
4. A. Biere. CaDiCaL, Lingeling, Plingeling, Treengeling, YalSAT Entering the SAT Competition 2017 In *Proc. of SAT Competition 2017Solver and Benchmark Descriptions, ser. Dept. of Comp. Science Series of Publications B, vol. B-2017-1* **2017**.
5. G. Kwon and H. Jain. Optimized CNF Encoding for Sudoku Puzzles. In *Proc. of LPAR'06* **2006**.
6. Alan M. Frisch, Timothy J. Peugniez, Anthony J. Doggett, and Peter Nightingale. Solving non-Boolean satisfiability problems with stochastic local search: A study of encodings. In *Journal of Automated Reasoning* **2005**.
7. Alan M. Frisch and Timothy J. Peugniez. Solving non-Boolean satisfiability problems with stochastic local search. In *Proc. of the Seventeenth Int. Joint Conf. on Articial Intelligence* **2001**.
8. Carsten Sinz. Towards an optimal CNF encoding of Boolean cardinality constraints. In*Proc. of the 11th Intl. Conf. on Principles and Practice of Constraint Program-ming* **2005**.