electronics

MDPI

Communication

Fractional Encoding of At-Most-K Constraints on SAT

Miki Yonekura [1] and Shunji Nishimura [2,*]

[1] Electrical, Electronics Information Engineering Major, Advanced Course, National Institute of Technology, Oita College, Oita 870-0152, Japan; yonekura.miki.yn4@is.naist.jp
[2] Information Engineering Department, National Institute of Technology, Oita College, Oita 870-0152, Japan
* Correspondence: s-nishimura@oita-ct.ac.jp

Abstract: The satisfiability problem (SAT) in propositional logic determines if there is an assignment of values that makes a given propositional formula true. Recently, fast SAT solvers have been developed, and SAT encoding research has gained attention. This enables various real-world problems to be transformed into SAT and solved, realizing a solution to the original problems. We propose a new encoding method, Fractional Encoding, which focuses on the At-Most-K constraints—a bottleneck of computational complexity—and reduces the scale of logical expressions by dividing target variables. Furthermore, we confirm that Fractional Encoding outperforms existing methods in terms of the number of generated clauses and required auxiliary variables. Hence, it enables the efficient solving of real-world problems like planning and hardware verification.

Keywords: At-Most-K constraints; satisfiability problem; logical expression

1. Introduction

The satisfiability problem (SAT) in propositional logic is the determination of whether there exists an assignment of values that makes a given propositional formula true. Recently, very fast SAT solvers have been developed, and the study of SAT encoding has attracted attention. Problems such as planning, hardware verification, software verification, and scheduling are transformed into SAT and solved using a SAT solver, realizing a solution to the original problem [1,2]. In recent years, there has been active research into SAT encoding for optimal Clifford circuits [3,4], expanding the application range of SAT.

Real-world problems encoded into SAT are composed of various constraints, among which the At-Most-K constraints often become the bottleneck of computational complexity [2]. Regarding state-of-the-art SAT papers, Timm et al. used SAT for the verification of multi-agent systems [5]. In this situation, an efficient encoding method is needed, especially for At-Most-K constraints for large K. To date, various encoding methods for At-Most-K constraints have been proposed. Frisch et al. proposed Binary Encoding, which performs efficient encoding by assigning domains to each target variable [6,7]. Sinz et al. introduced Counter Encoding, which operates efficiently by referring to sequential counting circuits [8]. There is still room for improvement in these methods in terms of suppressing the scale of logical expressions. Therefore, in this study, we propose a new method, Fractional Encoding, which suppresses the scale of logical expressions by dividing the target variables. The performance of the encoding is evaluated by the number of clauses generated and the number of auxiliary variables required. As a result, it has been confirmed that the proposed Fractional Encoding performs better than existing methods in terms of the number of clauses generated and the number of auxiliary variables.

The starting point of this study is the preliminary report [9], in which the idea of Fractional Encoding is presented. There is also related research [10] that is based on the same idea and proposed Approximate Encoding of At-Most-K constraints. While Fractional Encoding in this paper has fine-tuning variables to be described later, Approximate Encoding does not, thus Approximate Encoding cannot cover all possible solutions for At-Most-K constraints.

Citation: Yonekura, M.; Nishimura, S. Fractional Encoding of At-Most-K Constraints on SAT. *Electronics* **2023**, *12*, 3211. https://doi.org/10.3390/electronics12153211

Academic Editors: Shin'ya Obara and Eiji Takada

Received: 28 June 2023
Revised: 21 July 2023
Accepted: 23 July 2023
Published: 25 July 2023

1.1. 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 [11], which is well-known, and CaDiCaL [12], 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.

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 [13].

1.2. 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, Pairwise Encoding with at most one true variable and three target variables (x_1, x_2, x_3) is

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

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. The two methods are described below.

1.3. Binary Encoding

Binary encoding was originally introduced by Frisch et al. [6,7]. The encoding introduces new variables $B_1, \ldots, B_{\lceil \log_2 n \rceil}$. It then associates with each x_i a unique bit string $s_i \in \{1,0\}^{\lceil \log_2 n \rceil}$. The binary encoding of At-Most-One constraint is

$$\bigwedge_{i=1}^{n} \bigwedge_{j=1}^{\lceil \log_2 n \rceil} \neg x_i \vee \varphi(i,j) \tag{2}$$

where $\varphi(i,j)$ denotes B_j if the jth bit of s_i is 1 and otherwise denotes $\neg B_j$. The binary encoding can extend to the At-Most-K constraint. As before, associate with each x_i a unique bit string $s_i \in \{1,0\}^{\lceil \log_2 n \rceil}$. The encoding introduces new variables $B_{i,g}$ $(1 \leq i \leq K, 1 \leq g \leq \lceil \log_2 n \rceil)$, which are essentially K copies of the previous B variables. The binary encoding of At-Most-K constraint is

$$\bigwedge_{i=1}^{n} \bigwedge_{g=1}^{K} \bigwedge_{j=1}^{\lceil \log_2 n \rceil} \neg x_i \vee \varphi(i,g,j)$$

where $\varphi(i,g,j)$ denotes $B_{g,j}$ if the jth bit of s_i is 1 and otherwise denotes $\neg B_{g,j}$.

1.4. Counter Encoding

Sinz introduced an encoding that works by encoding a circuit that sequentially counts the number of x_i that are true [8]. For each $1 \leq i \leq n$ there is a register whose value is constrained to contain the number of $x_1, \ldots, x_i$ that are true. Each register maintains its count in base one and hence uses K bits to count to K. Thus, the encoding introduces the

new variables $R_{i,j}$, $1 \leq i \leq n$, $1 \leq j \leq K$, where each $R_{i,j}$ represents the ith bit of register j. The clauses of the encoding are as follows.

$$\bigwedge_{i=1}^{n-1} \neg x_i \vee R_{i,1} \tag{3}$$

$$\bigwedge_{j=2}^{K} \neg R_{1,j} \tag{4}$$

$$\bigwedge_{i=2}^{n-1} \bigwedge_{j=1}^{K} \neg R_{i-1,j} \vee R_{i,j} \tag{5}$$

$$\bigwedge_{i=2}^{n-1} \bigwedge_{j=2}^{K} \neg x_i \vee \neg R_{i-1,j-1} \vee R_{i,j} \tag{6}$$

$$\bigwedge_{i=2}^{n} \neg x_i \vee \neg R_{i-1,K} \tag{7}$$

Formula (3) states that if x_i is true then the first bit of register i must be true. Formula (4) ensures that in the first register only the first bit can be true. Formulas (5) and (6) together constrain each register i ($1 < i < n$) to contain the value of the previous register plus x_i. Finally, (7) asserts that there cannot be an overflow on any register as it would indicate that more than K variables are true.

The encoding method proposed in this study, "Fractional encoding", also suppresses the number of clauses better than pairwise encoding by introducing auxiliary variables. The remainder of this paper is organized as follows: Section 2 presents the method. Section 3 presents the results. Additionally, Section 4 contains a discussion, and concluding remarks are provided in Section 5.

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 $AtMost\ K/n\,(a\ set\ of\ target\ variables)$. The overall flow of the research is shown in Figure 1.

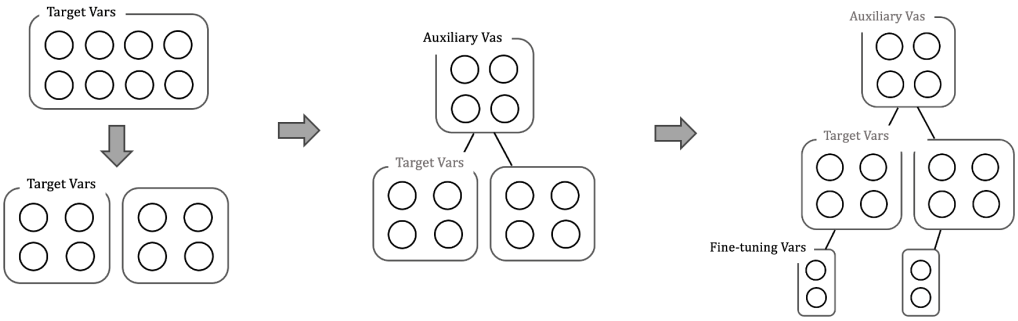

Figure 1. The proposed method solves the reduction of the solution space caused by dividing the target variables through two stages of additional variables.

2.1. 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.

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

$$\underbrace{{}_8C_5}_{56\ \text{clauses}} \xrightarrow{Split\ in\ two} \underbrace{{}_4C_3 + {}_4C_3}_{8\ \text{clauses}} \quad (9)$$

The number of clauses in Pairwise Encoding is calculated by ${}_nC_{K+1}$ and can be reduced from 56 to 8 as shown in (9). However, this example does not lead to *assignment* $= x_1, x_2, x_3, x_5$ such that 3 variables in set $x_1, \ldots, x_4$ and 1 variable in set $x_5, \ldots, x_8$ are true. As a result, a simple splitting of the set greatly reduces the number of possible combinations of variables that can be solved (solution space). Figure 2 illustrates this phenomenon in the "split target vars" section.

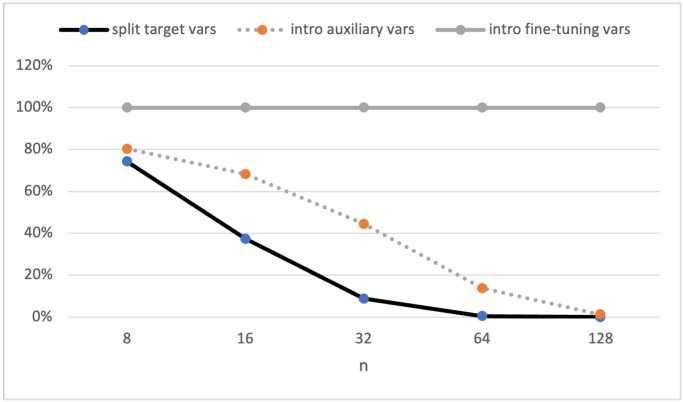

Figure 2. This paper demonstrates the reduction of the solution space at each step considered. It shows the coverage when simply splitting the target variables, the coverage after introducing the auxiliary variables, and the coverage after introducing the fine-tuning variables, which will be discussed later.

2.2. 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 Figure 3. 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 decreasing. As an example, the splitting of target variables for the $AtMost\ 8/16(x_1, \ldots, x_{16})$ constraint is shown below.

$$\begin{aligned} &AtMost\ 8/16(x_1, \ldots, x_{16}) \\ &\xrightarrow{Split\ in\ four} AtMost\ k_1/4(x_1, \ldots, x_4) \\ &\quad + AtMost\ k_2/4(x_5, \ldots, x_8) \\ &\quad + AtMost\ k_3/4(x_9, \ldots, x_{12}) \\ &\quad + AtMost\ k_4/4(x_{13}, \ldots, x_{16}) \end{aligned} \quad (10)$$

In (10), the 16 target variables are split into four subsets of $G_4 = \{x_1, \ldots x_4\}, \ldots, G_7$. $k_1, \ldots, k_4$ are dynamically determined values whose total value is less than or equal to K, as shown below.

$$k_1 + k_2 + k_3 + k_4 = K \tag{11}$$

The variables belonging to $G_1, \ldots, G_3$ 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 3 to realize (10) and (11). 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.

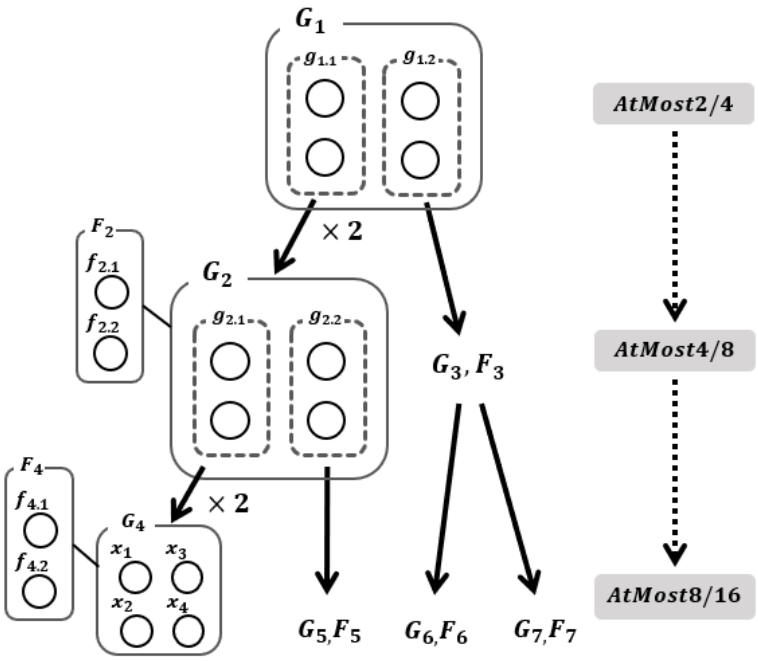

Figure 3. This figure shows the tree structure for generating $AtMost\ 8/16(x_1, \ldots, x_{16})$ using Fractional Encoding. The bottom layer serves as the target variables, and the other layers play the role of auxiliary variables. Each layer has a group G_i containing four variables, the number of 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.

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).

$$AtMost2(G_1) \tag{12}$$

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 4.

$$plus(i) : f_{i.1} \wedge f_{i.2} \tag{13}$$
$$minus(i) : \neg f_{i.1} \wedge \neg f_{i.2} \tag{14}$$
$$const(i) : f_{i.1} \oplus f_{i.2} \tag{15}$$

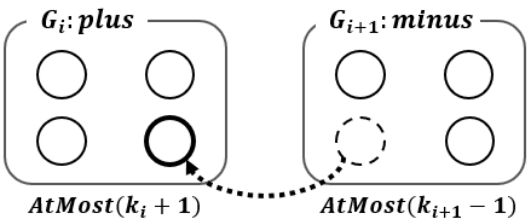

Figure 4. As shown in this figure, if the state is *plus*, the number of variables that can be true in the group is $+1$, and if *minus*, it is -1.

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.

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} Exactly0(g_{i.j}) \wedge const(2i + j) \Rightarrow AtMost0(G_{2i+j}) \tag{16}$$

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} (Exactly0(g_{i.j}) \wedge plus(2i + j)) \vee (Exactly1(g_{i.j}) \wedge minus(2i + j)) \Rightarrow AtMost1(G_{2i+j}) \tag{17}$$

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} Exactly1(g_{i.j}) \wedge const(2i + j) \Rightarrow AtMost2(G_{2i+j}) \tag{18}$$

$$\bigwedge_{i=1}^{p} \bigwedge_{j=0}^{1} (Exactly1(g_{i.j}) \wedge plus(2i + j)) \vee (Exactly2(g_{i.j}) \wedge minus(2i + j)) \Rightarrow AtMost3(G_{2i+j}) \tag{19}$$

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.

$$ExactrlyK \Leftrightarrow AtMostK \wedge AtLeastK \tag{20}$$

The leaf variables (target variables) generated by (16), (17), (18), and (19) 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.

$$Exactly0(g_{i.j}) \Rightarrow \neg minus(2i + j) \tag{21}$$

$$Exactly2(g_{i.j}) \Rightarrow \neg plus(2i + j) \tag{22}$$

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 $q = number\ of\ layers$.

$$\bigwedge_{m=1}^{q} AtMost2^m(f_{2^m.1} \ldots f_{2^{m+1}-1.2}) \tag{23}$$

2.3. 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.

$$Fix\ x_i\ as\ true : x_i \tag{24}$$

$$Fix\ x_i\ as\ false : \neg x_i \tag{25}$$

As shown below, when the 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 5 shows an example of three variables fixed.

$$Fix\ c\ true : \ AtMost\frac{K-c}{n-c} \tag{26}$$

$$Fix\ c\ false : \ AtMost\frac{K}{n-c} \tag{27}$$

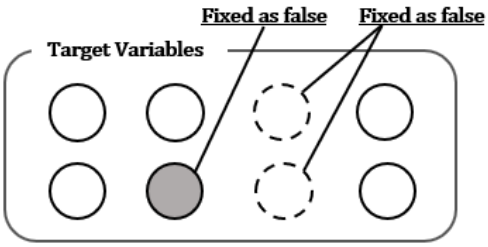

Figure 5. In the example, by fixing three variables, we transformed At-Most-4/8 to At-Most-3/5. By extending the possible patterns that can be generated in this way, more real-world problems can be solved.

2.4. 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} as shown in Figure 6, which is important in the clauses number calculation, will be called Local-Propagation.

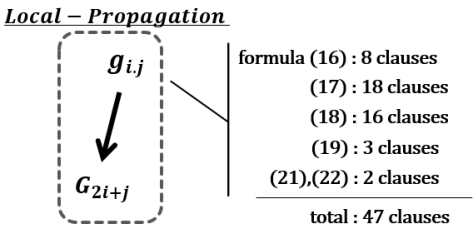

Figure 6. The number of clauses required when each equation required for Local-Propagation is converted to CNF. The total of these is the number of clauses required per Local-Propagation, which is 47.

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

$$\begin{aligned} clauses(AtMostK/n) = \\ clauses(AtMost\frac{K}{2^1}/\frac{n}{2^1}) + clauses(AtMost\frac{K}{2^2}/\frac{n}{2^2}) \\ \ldots + clauses(AtMost2/4) + clauses(Local\text{-}Propagations) \end{aligned} \quad (28)$$

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 (28); the second term shows the clauses (Local-Propagation) that are added when (28) is calculated recursively.

$$\begin{aligned} (2^r - 2) + \sum_{x=0}^{r-3} 2^x \times ((2^{r-1-x} - 2)) \\ = r \times 2^{r-1} - 2^{r-1} \\ = \frac{1}{2}K(\log_2 K - 1) \end{aligned} \quad (29)$$

$$clauses : 47 \times \frac{1}{2}K(\log_2 K - 1) \quad (30)$$

There are two types of auxiliary variables that are required for Fractional Encoding: 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 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.

$$6G - 2G_T - 4G_B \quad (31)$$

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.

$$\begin{aligned} 3K \log_2 K - K - (\log_2 K + 1) \\ = 2K(\log_2 K - 1) \end{aligned} \quad (32)$$

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

Table 1. Comparison of the number of clauses and auxiliary variables.

Method	Origin	Clauses	Auxiliary Vars
Pairwise	folklore	${}_nC_{k+1}$	0
Binary	Frisch [6,7]	$O(Kn \log_2 n)$	$O(Kn)$
Counter	Sinz [8]	$O(Kn)$	$O(Kn)$
Fractional	this paper	$O(K \log_2 K)$	$O(K \log_2 K)$

3. Results

3.1. Justification of the Proposed Method

In order to verify the validity of the proposed method, two verifications are 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 where 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 targeted 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.

3.2. Comparison with Conventional Methods

In order to compare 2/4-based Fractional Encoding with conventional methods, we examined the number of clauses (Figure 7), the number of auxiliary variables (Figure 8), and the total number of literals (Figure 9). These figures can be said to directly represent the differences in the order calculation tables of each method shown in Table 1.

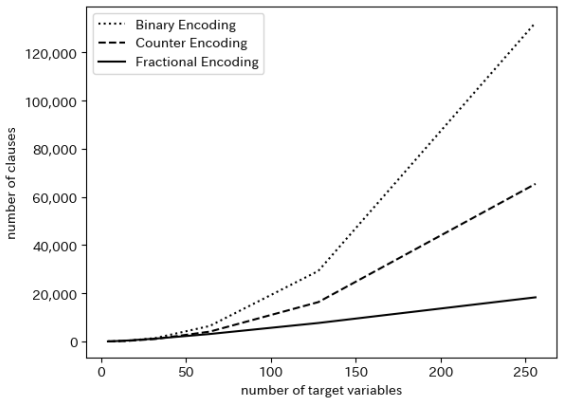

Figure 7. This graph shows the number of clauses in relation to the target variables. The two conventional methods show a significant increase in the number of clauses when the number of target variables exceeds 60. In contrast, Fractional Encoding succeeds in suppressing the number of clauses with a gradual increase.

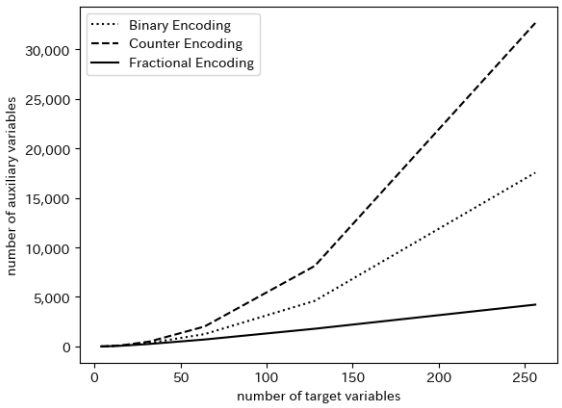

Figure 8. This graph shows the number of auxiliary variables for the target variables. Fractional Encoding also increases gradually and succeeds in suppressing the number of auxiliary variables compared to the conventional methods.

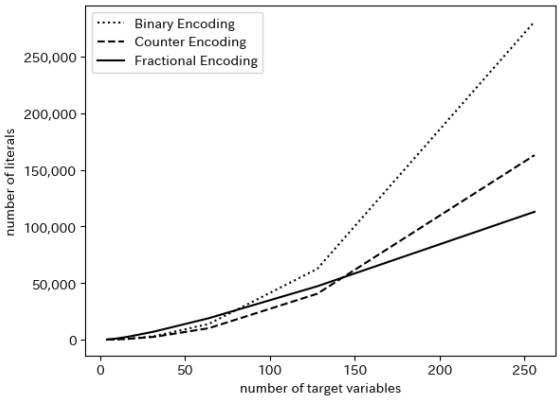

Figure 9. This graph shows the total number of literals for target variables. Fractional Encoding becomes superior to Binary Encoding when the number of target variables exceeds 70, and superior to Counter Encoding when the number of target variables exceeds 120.

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. Regarding the correction ability of Fractional Encoding, no verification has been performed yet, and it is necessary to verify the possibility of demonstrating superior performance even when overhead occurs due to "Pattern Extension with Variable Fixation" compared to existing methods.

In the near future, in an effort to generalize Fractional Encoding, we plan to examine methods for finding base fractions and verify the correction ability of Fractional Encoding.

5. Conclusions

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.

Author Contributions: Conceptualization, S.N.; methodology, M.Y. and S.N.; software, validation and writing, M.Y.; supervision, S.N. All authors have read and agreed to the published version of the manuscript.

Funding: APC was funded by [National Institute of Technology (KOSEN), Japan].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Frisch, A.M. SAT Encodings of the At-Most-k Constraint. Some Old, Some New, Some Fast, Some Slow. In Proceedings of the Ninth International Workshop of Constraint Modelling and Reformulation, St. Andrews, UK, 6 September 2010.
2. Bittner, P.M. SAT Encodings of the At-Most-k Constraint A Case Study on Conguring University Courses. In *SEFM 2019: Software Engineering and Formal Methods*; Springer Nature: Cham, Switzerland, 2019.
3. Young, J.M.; Bittner, P.M.; Walkingshaw, E.; Thüm, T. Variational satisfiability solving: Efficiently solving lots of related SAT problems. *Empir. Softw. Eng.* **2023**, *28*, 14. [CrossRef]
4. Schneider, S.; Burgholzer, L.; Wille, R. A SAT Encoding for Optimal Clifford Circuit Synthesis. In Proceedings of the 28th Asia and South Pacific Design Automation Conference (ASPDAC), Tokyo, Japan, 16–19 January 2023; pp. 190–195.
5. Timm, N.; Botha, J.; Jordaan, S. Max-SAT-based synthesis of optimal and Nash equilibrium strategies for multi-agent systems. *Sci. Comput. Program.* **2023**, *228*, 102946. [CrossRef]
6. Frisch, A.M.; Peugniez, T.J.; Doggett, A.J.; Nightingale, P.W. Solving non-Boolean satisfiability problems with stochastic local search: A study of encodings. *J. Autom. Reason.* **2005**, *35*, 143–179. [CrossRef]
7. Frisch, A.M.; Peugniez, T.J. Solving non-Boolean satisfiability problems with stochastic local search. In Proceedings of the Seventeenth International Joint Conference on Articial Intelligence, Seattle, WA, USA, 4–10 August 2001.
8. Sinz, C. Towards an optimal CNF encoding of Boolean cardinality constraints. In Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming, Sitges, Spain, 1–5 October 2005.
9. Yonekura, M.; Nishimura, S. Fractional Encoding of At-Most-K Constraint on SAT. In Proceedings of the 1st KOSEN Rearch International Symposium (KRIS), Tokyo, Japan, 1–2 March 2023; p. B-O-3-3.
10. Nishimura, S. Approximate-At-Most-k Encoding of SAT for Soft Constraints. In Proceedings of the 14th Pragmatics of SAT International Workshop (PoS2023), Alghero, Italy, 4–8 July 2023.
11. Een, N.; Sorensson, N. An extensible SAT-solver. In Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing, Santa Margherita Ligure, Italy, 5–8 May 2003.
12. Biere, A. CaDiCaL, Lingeling, Plingeling, Treengeling, YalSAT Entering the SAT Competition 2017. In *Proceedings of the SAT Competition 2017—Solver and Benchmark Descriptions*; Department of Computer Science Series of Publications B; University of Helsinki: Helsinki, Finland, 2017; Volume B-2017-1.
13. Kwon, G.; Jain, H. Optimized CNF Encoding for Sudoku Puzzles. In Proceedings of the 13th International Conference on Logic for Programming Artificial Intelligence and Reasoning, LPAR'06, Phnom Penh, Cambodia, 13–17 November 2006.