SAT変換に基づく制約ソルバーとその性能評価

田村 直之　丹生 智也　番原 睦則

本論文では，SAT 変換に基づく制約ソルバーである Sugar の概要とその性能評価結果について述べる．Sugar は，制約充足問題 (CSP)，制約最適化問題 (COP) および最大制約充足問題 (Max-CSP) を，命題論理の充足可能性判定問題 (SAT 問題) に変換し，MiniSat 等の高速な SAT ソルバーを用いて求解を行うシステムである．SAT 変換には，order encoding と名付けた新しい方法を用いており，従来広く用いられている direct encoding や support encoding よりも，多くの問題に対して高速な求解が可能である．本論文では，order encoding の説明を含めた Sugar の概要について述べた後，2008 年に開催された第 3 回国際 CSP ソルバー競技会および Max-CSP ソルバー競技会での結果を基に Sugar の性能評価結果を報告する．なお，Sugar は同競技会の 10 部門のうち 4 部門で第 1 位となった．

In this paper, we describe a SAT-based constraint solver named Sugar and its performance evaluation. Sugar can solve CSP (Constraint Satisfaction Problem), COP (Constraint Optimization Problem), and Max-CSP by encoding a given problem into a SAT problem and then solving the encoded SAT problem with an external efficient SAT solver (e.g. MiniSat). As for the SAT encoding, a new encoding method named order encoding is used, which is more efficient for various problems compared with the direct encoding and the support encoding which are widely used. This paper also describes the summary of the results at the 2008 Third International CSP/Max-CSP Solver Competitions. In the competitions, Sugar became the winner in four categories out of ten categories.

1　はじめに

命題論理の充足可能性判定問題 (SAT 問題) は，与えられた命題論理式を真にする値割り当てが存在するか否かを判定する問題である [3] [11] [10].

近年になって，SAT 問題を解くための非常に高速な SAT ソルバーが実現され [16]，プランニング，ハードウェア検証，ソフトウェア検証，スケジューリング等の問題について，それらを SAT 問題に変換した後，SAT ソルバーに解かせることにより，元の問題に対する求解を実現する SAT 変換 (あるいは SAT 符号化; SAT encoding) の研究が注目を集めている [14] [19] [5] [9] [17] [28] [15] [1]．たとえば，プランニング問題について，現在最速のソルバーの 1 つは，SAT 変換に基づく SATPLAN である [14]．この点で，SAT 問題は求解が困難な探索問題に対するアセンブリ言語としての位置付けがなされているといえる．

著者らは，整数上の線形の制約からなる制約充足問題に対して，order encoding と呼ぶ SAT 変換手法を提案している [26]．この SAT 変換方法は，Crawford と Baker によりジョブショップ・スケジューリング問題に適用され [5]，井上，宋，鍋島らにより論文 [9] [17] で研究された SAT 変換方法を，より一般の制約充足問題および制約最適化問題に適用可能なように拡張

A SAT-based Constraint Solver and its Performance Evaluation.

Naoyuki Tamura, 神戸大学情報基盤センター, Information Science and Technology Center, Kobe University.

Tomoya Tanjo, 神戸大学大学院工学研究科, Graduate School of Engineering, Kobe University.

Mutsunori Banbara, 神戸大学情報基盤センター, Information Science and Technology Center, Kobe University.

コンピュータソフトウェア, Vol.27, No.4 (2010), pp.183–196.
[ソフトウェア論文] 2008 年 10 月 20 日受付.

したものであり，未知だったオープンショップ・スケジューリング問題の最適値決定に成功する等，非常に有望な手法であることが示されている [25] [22].

本論文では，整数有限領域上の線形の制約充足問題 (CSP) および最大制約充足問題 (Max-CSP) を対象として，order encoding による SAT 変換方法，およびそれを実現した制約ソルバーである Sugar †1 [24] [27] [29] について述べ，第 3 回国際 CSP ソルバー競技会 (CSP solver competition) および Max-CSP ソルバー競技会 (以下，CSP/Max-CSP ソルバー競技会) [30] のベンチマーク問題に対する実行結果を元に性能評価を行う．

なお，CSP ソルバー競技会には，SAT ソルバーに MiniSat を用いた「Sugar+minisat」および PicoSAT を用いた「Sugar+picosat」の 2 ソルバーで参加し，Max-CSP ソルバー競技会には，SAT ソルバーに MiniSat を用いた「Sugar」および MiniSat のインクリメンタル探索機能を利用した「Sugar++」の 2 ソルバーで参加した．以下では，これらの 4 ソルバーを区別する必要のない場合，総称として Sugar を用いる．

2 制約充足問題

制約充足問題 (CSP, Constraint Satisfaction Problem) は，各変数に与えられたドメインから値を割り当て，与えられた制約のすべてを満たすことができるかどうかを判定する問題である [4]．すべての制約を満たす値割り当てが存在する場合，元の制約充足問題は充足可能 (satisfiable) であり，その値割り当てが解となる．値割り当てが存在しない場合，元の制約充足問題は充足不能 (unsatisfiable) である．また，制約充足問題の解を探索するプログラムを**制約ソルバー**と呼ぶ．

整数有限領域上の制約充足問題は，形式的には以下のように定義できる．

定義 1 (制約充足問題) 制約充足問題 (CSP) は以下を満たす組(V, Dom, C)である．

(1) V は整数変数の有限集合 $\{x_1, x_2, \ldots, x_n\}$ である．

(2) 関数 $Dom : V \to \mathcal{P}(\mathbf{Z})$ は，各変数の取り得る値集合 (ドメイン) を表す ($\mathcal{P}(\mathbf{Z})$ は整数 $\mathbf{Z}$ のベキ集合)．各変数 $x_i \in V$ について，$Dom(x_i)$ は $\mathbf{Z}$ の有限部分集合である．

(3) C は V 上の制約の有限集合 $\{C_1, C_2, \ldots, C_m\}$ であり，制約の連言を表す．

また，**最大制約充足問題** (Max-CSP) は，与えられた制約充足問題(V, Dom, C)について，同時に充足可能となる制約の個数を最大化する (すなわち，充足不能となる制約の個数を最小化する) 問題である．

制約には，算術論理演算等で条件が記述されている**内包的制約** (intensional constraint)，制約を満足する (あるいは制約に違反する) 点集合が陽に与えられている**外延的制約** (extensional constraint)，そして alldifferent 等に代表される**グローバル制約** (global constraint) がある．

CSP/Max-CSP ソルバー競技会では，内包的制約の記述において，論理演算として `and` (論理積), `or` (論理和)，`xor` (排他的論理和)，`iff` (同値)，`not` (否定)，`true` (真)，`false` (偽) が使用され，比較演算として `eq` (=)，`ne` ($\neq$)，`ge` ($\geq$)，`gt` ($>$)，`le` ($\leq$)，`lt` ($<$) が，算術演算として `add` (加算)，`sub` (減算)，`neg` (マイナス)，`mul` (乗算)，`div` (除算)，`mod` (剰余)，`pow` (ベキ乗)，`abs` (絶対値)，`min` (最小値)，`max` (最大値)，`if` (if 式) が使用された．また，グローバル制約としては，与えられたすべての変数が互いに異なることを表す `alldifferent` 制約，資源制約を表す `cumulative` 制約，リスト要素を表す `element` 制約，線形和を表す `weightedsum` 制約が使用された．

2.1 制約の記述例

CSP/Max-CSP ソルバー競技会では，制約充足問題は XML 形式で記述されているが，可読性が低いため，ここでは Sugar で採用している Lisp 風のリスト表現を用い，制約充足問題の記述例を説明する．

整数変数は，以下のように上限および下限を与えるか，あるいは要素を列挙して宣言する．

```
(int x 1 10)        ; x ∈ {1,2,...,10}
(int y (1 3 5..7)) ; y ∈ {1,3,5,6,7}
```

以下は，$(x + 2 \leq y) \vee (y + 3 \leq x)$ を表す内包的制

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

約の例である．

```
(or (<= (+ x 2) y) (<= (+ y 3) x))
```

次は，2 変数の外延的制約 r を定義し，利用している例であり，$(x, y) \in \{(1,3),(2,5),(3,7)\}$ という制約式を表す．

```
(relation r 2 (supports (1 3) (2 5) (3 7)))
(r x y)
```

ここで，supports は支持点集合を意味する．conflicts を用いた場合は違反点集合を意味し，$(x, y) \notin \{(1,3),(2,5),(3,7)\}$ という制約を表す．

次は，グローバル制約 alldifferent の例であり，x, y, z が互いに異なることを表す．

```
(alldifferent x y z)
```

3 SAT 問題と SAT ソルバー

命題論理の**充足可能性判定問題** (**SAT 問題**) は，与えられた命題論理式が充足可能であるか充足不能であるかを判定する問題であり，Cook により NP 完全であることが初めて示された問題でもある．

判定対象の論理式は，通常 **CNF** (Conjunctive Normal Form) で与えられる．すなわち，全体の論理式はいくつかの**節** (clause) の連言 (AND) であり，各節はいくつかの**リテラル** (literal，ブール変数またはその否定) の選言 (OR) となっている．

Davis らによって 1962 年に DPLL アルゴリズムが考案されて以来，SAT 問題を解くためのプログラムである **SAT ソルバー**の研究は近年長足の進歩を遂げており，最新の SAT ソルバーは数千万リテラルからなる問題を解くことができる．

SAT ソルバーには系統的に解を探索し充足可能/充足不能の双方を判定する系統的 SAT ソルバーと，確率的に解を探索し充足可能な場合のみを判定する確率的 SAT ソルバーの二種類が存在する．2002 年以降ほぼ毎年開催されている SAT 競技会 (SAT competition)[†2] では，DPLL アルゴリズムに基づいた系統的 SAT ソルバーが上位を占めている．

特に MiniSat[†3]は，2005 年 SAT 競技会での成功以降，高く評価されている系統的 SAT ソルバーである [7]．MiniSat は，C++で 1000 行程度のコードで，監視リテラル (watched literals) による高速な単位伝播 (unit propagation)，矛盾節 (conflict clause) の学習による探索空間の枝刈り，バックジャンプ法 (矛盾解析による後戻り)，決定変数選択のヒューリスティック (VSIDS)，リスタート (探索の再開) の導入等により，高速な探索を実現している．

また，一旦求解が終了した後も，結果が充足可能であれば，新たな仮定 (リテラルの集合で与えられる) を追加した下で続けて求解を行うインクリメンタル探索の機能を備えている．このインクリメンタル探索では，それ以前の探索中に学習した矛盾節をそのまま利用可能なため，類似した SAT 問題について繰り返し求解を行う場合に無駄な探索を削減できる．

PicoSAT[†4]は，2007 年 SAT 競技会の Industrial 部門において第 2 位 (充足可能な問題では第 1 位) となった SAT ソルバーであり，MiniSat をベースとして，頻繁なリスタートによる性能向上とより少ないメモリ消費を特徴としている [2]．

4 Order encoding による SAT 変換

CSP を SAT 問題に変換する方法としては，従来 direct encoding [6] [32]，support encoding [13] [8]，log encoding [12] などが知られている [20] [28]．

特に，**direct encoding** は，CSP の各整数変数 x および各整数定数 $a \in Dom(x)$ に対して，$x = a$ を表すブール変数 p_{xa} を用いる方法であり，制約ソルバーへの応用を含め広く用いられてきた．

しかし，この方法は整数の順序関係を利用していないため，変換後の SAT 問題が巨大となり，整数有限領域上の CSP に適用した場合，後述の order encoding に比べ性能が劣る．

著者らは，各整数変数 x と各整数定数 $a \in Dom(x)$ に対して，$x \leq a$ を意味するブール変数を用いる **order encoding** を提案している [26]．この方法は，Crawford と Baker によりジョブショップ・スケジューリング問題の SAT 変換に利用された方法 [5] をより一般に拡張したものである．

Order encoding は，整数の順序関係を自然に表現

†2 http://www.satcompetition.org

†3 http://minisat.se

†4 http://fmv.jku.at/picosat/

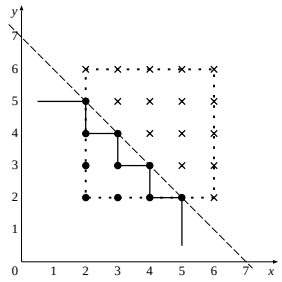

図 1　$x+y \leq 7$ の order encoding

した変換方法であるため，direct encoding と比較してよりコンパクトな変換結果となる．また，SAT ソルバーにおける単位伝播が CSP における範囲伝播に対応しており，高速な求解が実現可能な点が特徴である．

以下では，2 つの変数 $x, y \in \{2,3,4,5,6\}$ からなり，$x+y \leq 7$ を制約とする CSP を例に取り，direct encoding と order encoding の違いについて説明する．図 1 の黒点が，この CSP の解となる点である．

4.1　Direct encoding による SAT 変換例

Direct encoding は，CSP の各整数変数 x および各整数定数 $a \in Dom(x)$ に対して，$x = a$ を表すブール変数 p_{xa} を用いる [6] [32]．したがって，この例では変数毎に以下のような 5 個のブール変数を使用する．

$$p_{x2} \quad p_{x3} \quad p_{x4} \quad p_{x5} \quad p_{x6}$$

また各変数について，少なくとも 1 つの値を持つことを意味する at-least-one 節と，2 つ以上の値を持たないことを意味する複数の at-most-one 節を用意する．この例の場合，変数毎に以下のような 11 個の節が必要となる．

$$p_{x2} \vee p_{x3} \vee p_{x4} \vee p_{x5} \vee p_{x6}$$

$$\neg p_{x2} \vee \neg p_{x3} \quad \neg p_{x2} \vee \neg p_{x4} \quad \neg p_{x2} \vee \neg p_{x5}$$

$$\neg p_{x2} \vee \neg p_{x6} \quad \neg p_{x3} \vee \neg p_{x4} \quad \neg p_{x3} \vee \neg p_{x5}$$

$$\neg p_{x3} \vee \neg p_{x6} \quad \neg p_{x4} \vee \neg p_{x5}$$

$$\neg p_{x4} \vee \neg p_{x6} \quad \neg p_{x5} \vee \neg p_{x6}$$

一般には，ドメインのサイズを d とした場合，$O(d^2)$ 個の節を要する．

各制約は，制約に違反する点 (conflict point) をそれぞれ節に変換することで表現される．すなわち，点 $x_1 = a_1, \ldots, x_n = a_n$ が制約に違反する時，以下の節を追加する．

$$\neg p_{x_1 a_1} \vee \cdots \vee \neg p_{x_n a_n}$$

したがって，$x+y \leq 7$ の制約は，以下の 15 個の節で表わされる (各節は図 1 中の×の点に対応している)．

$$\neg p_{x2} \vee \neg p_{y6} \quad \neg p_{x3} \vee \neg p_{y5} \quad \neg p_{x3} \vee \neg p_{y6}$$

$$\neg p_{x4} \vee \neg p_{y4} \quad \neg p_{x4} \vee \neg p_{y5} \quad \neg p_{x4} \vee \neg p_{y6}$$

$$\neg p_{x5} \vee \neg p_{y3} \quad \neg p_{x5} \vee \neg p_{y4} \quad \neg p_{x5} \vee \neg p_{y5}$$

$$\neg p_{x5} \vee \neg p_{y6} \quad \neg p_{x6} \vee \neg p_{y2} \quad \neg p_{x6} \vee \neg p_{y3}$$

$$\neg p_{x6} \vee \neg p_{y4} \quad \neg p_{x6} \vee \neg p_{y5} \quad \neg p_{x6} \vee \neg p_{y6}$$

一般には，2 変数の線形制約について，$O(d^2)$ 個の節が必要となる．

4.2　Order encoding による SAT 変換例

Order encoding では，CSP の各整数変数 x および各整数定数 $a \in Dom(x)$ に対して，$x \leq a$ を表すブール変数 p_{xa} を用いる [26]．ただし，$a = \max(Dom(x))$ に対する p_{xa} は常に真なので不要である．したがって，この例では変数毎に以下のような 4 個のブール変数を使用する (p_{x6} は常に真のため不要)．

$$p_{x2} \quad p_{x3} \quad p_{x4} \quad p_{x5}$$

また各変数 x について，$Dom(x)$ 中の値に関して順序関係を表す節を用意する．この例の場合，変数毎に以下のような 3 個の節が必要となる．

$$\neg p_{x2} \vee p_{x3} \quad \neg p_{x3} \vee p_{x4} \quad \neg p_{x4} \vee p_{x5}$$

ここで $\neg p_{x2} \vee p_{x3}$ は,「$x \leq 2$ ならば $x \leq 3$」を表している．一般には，ドメインのサイズを d とした場合，$O(d)$ 個の節で良い．

この時，$p_{x2}, \ldots, p_{x5}$ の取り得る真理値の組み合わせは，表 1 に示すように 5 通りのみであり，それぞれ $x = 2$, $x = 3$, $\ldots$, $x = 6$ の場合に対応する．

各制約については，制約に違反する点ではなく違反する範囲 (conflict region) をそれぞれ節に変換することで実現できる．

$x+y \leq 7$ の制約は以下の 5 個の節となる．(図 1 の実線を参照)．

$$p_{y5} \quad p_{x2} \vee p_{y4} \quad p_{x3} \vee p_{y3} \quad p_{x4} \vee p_{y2} \quad p_{x5}$$

表 1 p_{xa} **の真理値表**

x の値	p_{x2}	p_{x3}	p_{x4}	p_{x5}
2	1	1	1	1
3	0	1	1	1
4	0	0	1	1
5	0	0	0	1
6	0	0	0	0

ここで $p_{x2} \vee p_{y4}$ は,「$x \leq 2$ または $y \leq 4$」であること，すなわち「$x > 2$ かつ $y > 4$」が制約に違反する領域であることを表している．一般に 2 変数の線形制約は，$O(d)$ 個の節で表現可能である．

より一般の線形制約については，次のようになる．制約 $\sum_{i=1}^{n} a_i\, x_i \leq c$ において，a_i を非零の整数定数，c を整数定数，x_i を互いに異なる整数変数とする．この時，制約は以下のような命題論理式に変換できる [26].

$$\bigwedge_{b_i} \bigvee_{i} (a_i\, x_i \leq b_i)^{\#}$$

ここで b_i は，$\sum_{i=1}^{n} b_i = c - n + 1$ を満たすように動くとし，変換 $()^{\#}$ は以下のように定義する．

$$(a\, x \leq b)^{\#} \quad \equiv \quad \begin{cases} x \leq \lfloor b/a \rfloor & (a > 0) \\ \neg\, (x \leq \lceil b/a \rceil - 1) & (a < 0) \end{cases}$$

ただし，$Dom(x)$ の最小値未満については偽に変換し，最大値以上については真に変換するものとする．

この方法で SAT 変換で生成される節数は，n 変数の線形制約について，$O(d^{n-1})$ となる．ただし後述のように，これは $O(n\, d^2)$ にできる．

4.3 SAT ソルバーの単位伝播との関係

Order encoding を用いた場合，SAT ソルバーでの基本動作である単位伝播 (unit propagation) が，整数変数のドメインに対する範囲伝播 (bounds propagation) に対応し，direct encoding 等の他の SAT 変換方法と比較して高速な推論を実現できる．

例えば，前節で例示した $x + y \leq 7$ の制約に対し，$x \geq 4$ すなわち $\neg p_{x3}$ を仮定すると，$p_{x3} \vee p_{y3}$ の節における単位伝播により直ちに p_{y3} すなわち $y \leq 3$ が得られる．

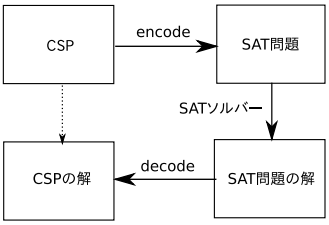

図 2 制約ソルバー Sugar

この推論が制約ソルバーにおける範囲伝播に対応し，矛盾節の学習による探索空間の枝刈り等の探索技法と組み合わさって，既存の制約ソルバーとは異なった特徴を持つ探索が実現されている．

5 制約ソルバー Sugar

制約ソルバー Sugar は，整数有限領域上の線形の制約充足問題を order encoding に基づき SAT 変換し，SAT ソルバーにより求解するシステムである (図 2 参照) [24] [27] [29]．SAT 変換部分は Java で記述され，SAT ソルバーとしては MiniSat, PicoSAT 等が利用可能である．

また，変換した SAT 問題を目的変数の範囲条件を変更しながら解くことにより，制約最適化問題 (COP) および最大制約充足問題 (Max-CSP) にも対応している．

Sugar では，与えられた CSP を，前処理により一旦 CSP の CNF 式に変換している．CSP の CNF 式におけるリテラルは，$\sum a_i x_i \leq b$ の形の線形制約，外延的制約，ブール変数，またはブール変数の否定のいずれかである．

なお，線形制約 $\sum a_i x_i \leq b$ の形になっていない比較式や算術式は，図 3 に示す方法で変換する．図中，“Expression” が元の式，“Replacement” が置換後の式，“Extra condition” が追加する制約である．また，E div c および E mod c は，式 E を正整数定数 c で割った商と剰余を表す．

また CNF 式への変換は，Tseitin 変換として良く知られているように，新しいブール変数を導入する方法で行っている．たとえば $p \vee (q \wedge r)$ は，新しいブール変数 p' を導入し，$(p \vee p') \wedge (\neg p' \vee q) \wedge (\neg p' \vee r)$ に変換する．この論理式の充足可能性は，元の論理式

Expression	Replacement	Extra condition
$E < F$	$E + 1 \leq F$	
$E = F$	$(E \leq F) \wedge (E \geq F)$	
$E \neq F$	$(E < F) \vee (E > F)$	
$\max(E, F)$	x	$(x \geq E) \wedge (x \geq F) \wedge ((x \leq E) \vee (x \leq F))$
$\min(E, F)$	x	$(x \leq E) \wedge (x \leq F) \wedge ((x \geq E) \vee (x \geq F))$
$\mathrm{abs}(E)$	x	$(x \geq E) \wedge (x \geq -E) \wedge ((x \leq E) \vee (x \leq -E))$
E div c	q	$(E = c\,q + r) \wedge (0 \leq r) \wedge (r < c)$
E mod c	r	$(E = c\,q + r) \wedge (0 \leq r) \wedge (r < c)$

図 3　$\sum a_i x_i \leq b$ 以外の式の変換

$p \vee (q \wedge r)$ の充足可能性と一致する [28].

その他，以下の方法の導入により実用的な SAT 変換型の制約ソルバーを実現している．

5.1　3 変数制約への置換

第 4 節で述べたように，線形制約 $\sum_{i=1}^{n} a_i x_i \leq b$ は，一般には $O(d^{n-1})$ 個の節に SAT 変換される．ここで，d はドメインのサイズを表す．

しかし，新しい整数変数を導入すれば，線形制約中の変数の個数を 3 個以下にできるため，線形制約 $\sum_{i=1}^{n} a_i x_i \leq b$ は，$n \geq 4$ の場合でも $O(n\,d^2)$ 個の節に SAT 変換できる．

たとえば $a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4 \leq b$ の場合，新しい整数変数 y を導入し，元の線形制約を $a_1x_1 + a_2x_2 + y \leq b$ に置き換えた上，2 個の新しい制約 $a_3x_3 + a_4x_4 - y \leq 0$, $-a_3x_3 - a_4x_4 + y \leq 0$ を追加すれば良い．この方法を繰り返すことによって，任意の線形制約について，変数の個数を 3 個以下にできる．

5.2　制約伝播によるドメインの縮小

ドメインのサイズを d とした時，d が非常に大きい場合は多数の節になる．そこで Sugar では，通常の制約ソルバーで行われているものと同様の制約伝播を行い，SAT 変換の前にドメインの縮小を行っている．たとえば，CSP ソルバー競技会のベンチマーク問題の 1 つである `FISCHER11-6-fair` については，制約伝播により 5.5 億以上の無駄なドメイン値が削除される．

5.3　グローバル制約の変換

グローバル制約については，基本的にはそれぞれの定義に従い同等の制約式に変換した後，SAT 変換を行っている．

ただし，与えられた n 個の整数変数 $x_1, \ldots, x_n$ が互いに異なることを意味する alldifferent 制約については，まず，定義に従って $\bigwedge_{i<j}(x_i \neq x_j)$ なる制約に変換し，さらに $\neg\bigwedge(x_i < lb + n - 1)$ および $\neg\bigwedge(x_i > ub - n + 1)$ という制約を追加する (ただし ub, lb は整数変数 $x_1, x_2, \ldots, x_n$ の上下限を表す)．これは，互いに異なる n 変数が $n-1$ のサイズのドメインに入らないことを表す鳩の巣原理 (pigeonhole principle) の条件を追加したことに相当し，大幅な性能向上を実現している [23].

alldifferent 制約についてはこれまで様々な整合性アルゴリズムが提案されている [31]．それらの内，範囲整合性 (bounds consistency) アルゴリズムは，n 個の整数変数が取り得るすべての範囲について鳩の巣原理による条件を利用する方法である．しかし，そのまま SAT 変換した場合，膨大な節数となる．

Sugar で用いている上記の方法は，最大の範囲についてのみ鳩の巣原理を利用することで，追加する節数を 2 つだけにし，効率的な変換を実現している点が特徴となっている．

5.4　COP への対応

制約最適化問題 (COP) は，与えられた CSP (V, Dom, C)および目的変数 $v \in V$ について，目的変数の値を最小あるいは最大にする解を求める問題である．

COP の解は，目的変数の範囲を 2 分探索等の手法

2	9	4
7	5	3
6	1	8

x_1	x_2	x_3
x_4	x_5	x_6
x_7	x_8	x_9

図 4 3×3 **の魔方陣**

で狭めながら複数の CSP を解くことで求められる．

しかし，目的変数の範囲の異なる CSP について各々SAT 変換を行うのは効率が悪い．

そこで Sugar では，CSP の SAT 変換は最初の 1 回のみ行い，作成された SAT 問題ファイルの最後に目的変数の範囲を表す SAT 節を追加することで，COP の解の探索を実現している．

ところが，この方法でも目的変数の範囲の変更のたびに SAT ソルバーを起動する必要がある．

Sugar を改良した Sugar++は，MiniSat のプログラムを改良しインクリメンタル探索機能を利用することにより，MiniSat プロセスの起動を一度だけにし，探索の連続実行を実現している [29]．探索の連続実行により，それまでの探索中に学習した矛盾節が再利用可能となり無駄な探索が削減できる．この方法は，鍋島らが論文 [17] で提案した手法と同様のものである．

5.5 Max-CSP への対応

最大制約充足問題 (Max-CSP) は，COP に変換することで求解を行える．

Sugar では，C_1，C_2，…，C_m を制約とする Max-CSP が与えられた時，新しい整数変数 $p \in \{0, 1, \ldots, m\}$ および $p_i \in \{0, 1\}$ $(i = 1, 2. \ldots, m)$ を導入し，下記の制約の下で目的変数 p の値を最小化する COP を構成する．

$$p \geq \sum_{i=1}^{m} p_i$$
$$(p_i > 0) \vee C_i \quad (i = 1, 2, \ldots, m)$$

ここで，p_i は制約 C_i が満たされない場合のペナルティを表している．

上記の最適解が元の Max-CSP の最適解となる．

5.6 Sugar での変換例

本節では，Sugar の実行例を紹介する．

```
(int x1 1 9)
(int x2 1 9)
(int x3 1 9)
(int x4 1 9)
(int x5 1 9)
(int x6 1 9)
(int x7 1 9)
(int x8 1 9)
(int x9 1 9)
(alldifferent x1 x2 x3 x4 x5 x6 x7 x8 x9)
(= (+ x1 x2 x3) 15)
(= (+ x4 x5 x6) 15)
(= (+ x7 x8 x9) 15)
(= (+ x1 x4 x7) 15)
(= (+ x2 x5 x8) 15)
(= (+ x3 x6 x9) 15)
(= (+ x1 x5 x9) 15)
(= (+ x3 x5 x7) 15)
```

図 5 3×3 **の魔方陣の CSP**

3×3 の魔方陣は，1 から 9 の数字を 3 行 3 列のマスに配置し，各行，各列および 2 つの対角線について配置されている数字の和がいずれも 15 となるようにする問題である．図 4 の左は解の 1 つを表している．

これを制約充足問題として定式化するために，図 4 の右側のように整数変数 x_i を各マスに割り当て，各 x_i のドメインを $\{1, 2, \ldots, 9\}$ と定める．必要な制約は，各マスの数字が互いに異なることを表す alldifferent 制約，および各行，各列および 2 つの対角線の和が 15 に等しいことを表す $x_1 + x_2 + x_3 = 15$ 等である．

図 5 に，Sugar への入力として記述した問題を示す．この入力ファイルに対して Sugar を実行すると，SAT 問題への変換，SAT ソルバーの起動，SAT ソルバーの結果の逆変換が行われ，以下のように元の問題に対する値割当て結果が表示される．

```
s SATISFIABLE
a x1    2
a x2    9
a x3    4
a x4    7
a x5    5
a x6    3
a x7    6
a x8    1
a x9    8
a
```

なお，詳細については「付録 A 3×3 魔方陣の Sugar による SAT 変換例」を参照されたい．

6 性能評価

6.1 CSP/Max-CSP ソルバー競技会

CSP/Max-CSP ソルバー競技会[†5]は，2008 年 6 月に開始され同年 9 月に結果が発表された [30].

提出された制約ソルバーは，主催者の用意した実行環境で，審判によって定められたそれぞれ 5 部門のベンチマーク問題に対して実行され，性能が評価された.

実行環境は以下の通りである.

- CPU: Xeon 2GHz, 2MB cache, 32-bits mode
- CPU 時間制限: 30 分 (CSP), 60 分 (Max-CSP)
- メモリ制限: 900MB

性能は以下の方法で比較され，各部門における順位付けが行われた.

- 上記実行環境で解けた問題の個数による比較
  - CSP については，`SATISFIABLE` または `UNSATISFIABLE` を解答した問題の個数
  - Max-CSP については，最適値を求めた問題の個数
- 解けた問題の個数が等しい場合は，CPU 時間による比較
- 一問でも間違った解答を行ったソルバーは，その部門で失格となり，評価の対象とならない.

表 2 および 3 に，CSP/Max-CSP ソルバー競技会における部門名，問題数，その部門に参加したソルバー数を示す．CSP ソルバー競技会については合計で 14 チーム，24 ソルバーの参加，Max-CSP ソルバー競技会では 4 チーム，8 ソルバーの参加だった (各チームは 2 ソルバーまでを参加登録できる).

各部門は，以下に示すようなベンチマーク問題から構成されている (表 4 参照).

- **2-ARY-EXT 部門**: 2 変数間の外延的制約からなる問題．ほとんどが乱数生成されたランダム CSP の問題である.
- **2-ARY-INT 部門**: 2 変数間の内包的制約および 2 変数間の外延的制約からなる問題．ショップ・スケジューリング，周波数割当，グラフ彩色，クイーン配置等の問題が含まれる.
- **N-ARY-EXT 部門**: 多変数間の外延的制約からなる問題．ランダム CSP，クロスワード等の問題が含まれる.
- **N-ARY-INT 部門**: 多変数間の内包的制約および外延的制約からなる問題．有界モデル検査，実時間相互排除プロトコル検証，マルチナップサック，擬似ブール制約，ゴロム定規，ソーシャルゴルファー等の問題が含まれる.
- **GLOBAL 部門**: グローバル制約および多変数間の内包的制約，外延的制約からなる問題．ラテン方陣，時間割作成等の問題が含まれる.

また，各部門の問題は以下のシリーズに分類されている.

- **REAL** (Real-World instances): 現実世界の応用例からなる問題.
- **PATT** (Patterned instances): 一定のパターンに従った問題 (ランダム生成を含む).
- **ACAD** (Academic instances): ランダム生成を含まない学術的な問題.
- **QRND** (Quasi-random instances): 小規模の構造を含みランダムに生成された問題.
- **RAND** (Random instances): 純粋にランダムに生成された問題.
- **BOOL** (Boolean instances): ブール変数 (0-1 変数) のみを含む問題.

表 2 CSP ソルバー競技会

部門名	問題数	ソルバー数
2-ARY-EXT	635	23
2-ARY-INT	696	22
N-ARY-EXT	704	24
N-ARY-INT	716	22
GLOBAL	556	17

表 3 Max-CSP ソルバー競技会

部門名	問題数	ソルバー数
2-ARY-EXT	534	8
2-ARY-INT	276	6
N-ARY-EXT	278	8
N-ARY-INT	109	6
GLOBAL	98	2

†5 `http://www.cril.univ-artois.fr/CPAI08/`

表 4 CSP/Max-CSP ソルバー競技会の部門

	2 変数		多変数		
部門名	外延的	内包的	外延的	内包的	グローバル
2-ARY-EXT	○				
2-ARY-INT	○	○			
N-ARY-EXT	○		○		
N-ARY-INT	○	○	○	○	
GLOBAL	○	○	○	○	○

表 5 CSP ソルバー競技会での Sugar ソルバーの結果

	Category	Rank	#Solved	% of VBS
Sugar+minisat	2-ARY-EXT	14	470	76%
	2-ARY-INT	11	484	76%
	N-ARY-EXT	15	370	61%
	N-ARY-INT	12	486	74%
	GLOBAL	4	405	84%
Sugar+picosat	2-ARY-EXT	15	443	71%
	2-ARY-INT	10	486	77%
	N-ARY-EXT	16	347	57%
	N-ARY-INT	13	481	73%
	GLOBAL	1	424	85%

6.2 CSP ソルバー競技会の結果

CSP ソルバー競技会には，SAT ソルバーとして MiniSat を用いた Sugar+minisat と，PicoSAT を用いた Sugar+picosat の 2 ソルバーで参加した [27].

表 5 に，CSP ソルバー競技会の各部門における Sugar ソルバーの結果を示す．“Rank” はその部門における順位，“#Solved” は解けた問題数，“% of VBS” は VBS (Virtual Best Solver) に対しての解いた問題数の割合を示す．VBS は，参加全ソルバーの最も良い結果を統合した仮想的なソルバーである．

Sugar ソルバーは，最も広い範囲の制約からなる GLOBAL 部門において，第 1 位および第 4 位という非常に優れた成績だった．

GLOBAL 部門以外の第 1 位は，すべて CPhydra というポートフォリオ型のソルバーであった．CPhydra は，内部に複数の制約ソルバーを持っており，CSP から抽出した特徴値からそれらの複数の制約ソルバーを実行する計画を作成し，その計画に基づいて内部の制約ソルバーを動作させ解を求めている [18]．今回の CPhydra は，競技会に同時参加した Mistral, Choco, Abscon の 3 種類の制約ソルバーを用い，前回競技会のベンチマーク問題について，事例ベース推論を用いて事前に学習を行っている．

外延的制約が中心の問題の場合，制約には整数の順序関係が現れておらず，order encoding による SAT 変換が有効に働く場合が少ない．2-ARY-EXT および N-ARY-EXT の部門において，Sugar がやや低い順位となっているのは，このことが原因と考えられる．

2-ARY-INT 部門の結果について，第 1 位の CPhydra (597 問解答) と Sugar (486 問解答) を比較した所，グラフ彩色問題で 31 問，周波数割当問題で 72 問少ない解答数となっており，これらが差のほとんどを占めていた．グラフ彩色問題の制約はすべて等号否定 ($\neq$) であり，整数の順序関係が現れておらず，order encoding による SAT 変換型ソルバーに不向きな問題といえる．周波数割当問題については，SAT 変換した際に SAT 問題の規模が大きくなりすぎたためメモリオーバーとなっているものが多く，今後に課題を残していることがわかった．

N-ARY-INT 部門で，同様に第 1 位の CPhydra (569 問) と Sugar (486 問) を比較した所，クロスワード問題で 58 問，Primes 問題で 28 問少なく，差の大きな要因となっていた．クロスワード問題は外延的制約も多数含まれている問題であり，やはり Sugar に不向きな問題といえる．Primes 問題は比較的大きな素数を係数とする線形制約を制約としている問題であり，

表 6 GLOBAL 部門での解けた問題数の比較

Series		Sugar	CPhydra	Mistral	CaSPER	Choco
ACAD: BIBD	(83)	78*	70	67	57	51
ACAD: Costas Array	(11)	8	9	9	9	9
ACAD: Latin Square	(10)	9*	5	5	6	5
ACAD: Magic Square	(18)	8	8	8	16	6
ACAD: NengFa	(3)	3*	3	3	2	3
ACAD: Orthogonal Latin Square	(9)	3*	2	2	3	2
ACAD: Perfect Square Packing	(74)	53*	52	41	44	49
ACAD: Pigeons	(19)	19*	19	19	19	19
ACAD: Quasigroup Existence	(35)	29	28	28	30	28
BOOL: Pseudo-Boolean	(100)	70*	44	40	69	49
PATT: BQWH	(20)	20*	20	20	20	20
PATT: Cumulative Job-Shop	(10)	4*	2	2	2	1
PATT: RCPSP	(78)	78*	78	78	70	73
REAL: Cabinet	(40)	0	40	40	40	40
REAL: Timetabling	(46)	42*	40	41	10	3
TOTAL	(556)	424*	420	403	397	358

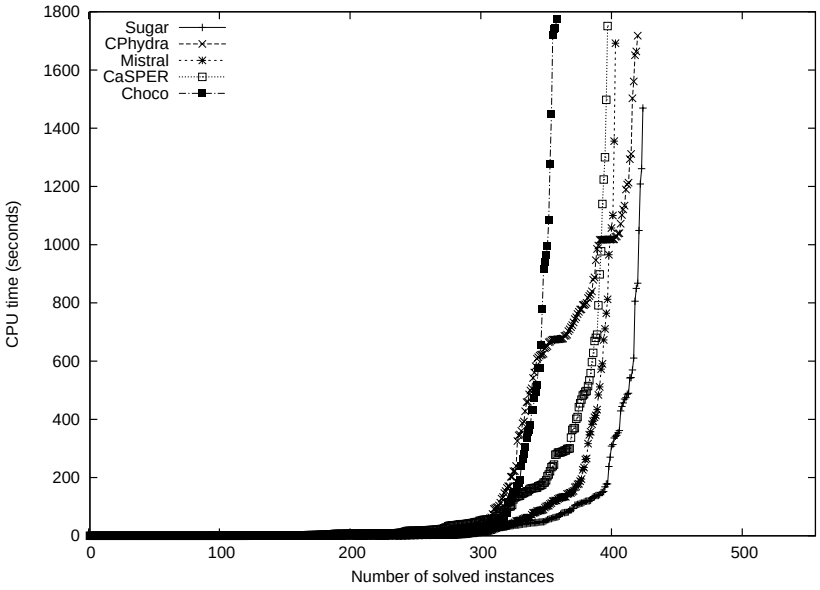

図 6 GLOBAL 部門: 制限時間内に解けた問題数

SAT 問題の規模が大きくなりメモリオーバーとなっているものが多く見られた.

最後に，GLOBAL 部門についての結果の詳細を表 6 に示し，図 6 に上位のソルバーの解いた問題数と CPU 時間のグラフを示す．GLOBAL 部門での順位は，第 1 位から第 9 位まで順に Sugar+picosat (424 問), CPhydra k_40 (420 問), CPhydra k_10 (419 問), Sugar+minisat (405 問), Mistral-prime (403 問), CaSPER zito (397 問), CaSPER zao (390 問), Mistral-option (383 問), Choco 2_dwdeg (358 問) であった．表 6 および図 6 では同一チームのソルバーについては上位のものだけを示している．また，表中 “Series” は問題のシリーズ名と総問題数である．また，Sugar 欄中の * 印は表中のソルバーのうちで最も良い結果であること表す.

Sugar は，GLOBAL 部門のほとんどのシリーズで

表 7 Max-CSP ソルバー競技会での Sugar ソルバーの結果

	Category	Rank	#Solved	% of VBS
Sugar	2-ARY-EXT	2	240	55%
	2-ARY-INT	1	101	98%
	N-ARY-EXT	2	118	69%
	N-ARY-INT	1	39	93%
	GLOBAL	1	65	100%
Sugar++	2-ARY-EXT	3	229	52%
	2-ARY-INT	2	99	96%
	N-ARY-EXT	3	118	69%
	N-ARY-INT	2	39	93%
	GLOBAL	2	50	77%

最も優れた性能を示した．しかし，2 位の CPhydra および他のソルバーと比較すると，$\{0, 1610\}$ 等の 2 値のドメインを持つ変数の線形和が使用されている Cabinet 問題が全く解けていなかった．2 値のドメインを持つ変数自体は，order encoding では 1 つのブール変数として効率良く変換される．しかし，それらの線形和を変換する際に新しい整数変数が導入され，そのドメインが大きくなるため，SAT 問題の規模も増大しメモリオーバーとなっていた．この点も今後の重要な検討課題である．

6.3 Max-CSP ソルバー競技会の結果

Max-CSP ソルバー競技会には，SAT ソルバーとして MiniSat を用いた Sugar と，MiniSat を改良しインクリメンタル探索機能を利用できるようにした Sugar++の 2 ソルバーで参加した [27] [29].

表 7 に，Max-CSP ソルバー競技会の各部門における Sugar ソルバーの結果を示す.

参加したソルバーは多くはないが，Sugar および Sugar++ソルバーは，2-ARY-INT, N-ARY-INT, GLOBAL 部門で第 1 位と第 2 位，2-ARY-EXT, N-ARY-EXT 部門で第 2 位と第 3 位という非常に優れた成績であった．ただし，GLOBAL 部門には Sugar と Sugar++以外のソルバーは参加していない.

2-ARY-EXT および N-ARY-EXT 部門の第 1 位は，それぞれ toulbar2 および toulbar2/BTD という外延的制約のみを対象としたソルバーである [21]．なお，toulbar2 は N-ARY-EXT 部門で，toulbar2/BTD は 2-ARY-EXT 部門で失格になったため，それらの部門での順位は与えられていない.

6.4 考察

CSP ソルバー競技会で，参加 24 ソルバー中 9 ソルバーがいずれかの部門で失格となっていることからもわかるように，高性能かつ信頼性の高い制約ソルバーを開発することは簡単な仕事ではない.

Sugar は，競技会における数千問のベンチマーク問題に対し間違った解答を行うことがなく，その点で性能および信頼性の高い制約ソルバーといえる.

以下では，これまでの記述と重複する点もあるが，CSP/Max-CSP ソルバー競技会の結果について考察事項をまとめる.

- Sugar で用いている order encoding は，広く用いられている direct encoding と比較して，よりコンパクトな SAT 変換を実現している．しかし，大規模な問題については，依然として変換後の SAT 問題が巨大となり，競技会の実行環境でメモリオーバーとなっていた．64 ビット CPU やより大きなメモリの利用が一般的になれば，自然に解消される問題ともいえるが，変換方法の工夫により解決できる可能性もあり，今後の重要な研究課題の 1 つである.
- グローバル制約について，alldifferent 制約への鳩の巣原理の導入以外には特別な処理を行っていないにもかかわらず，CSP ソルバー競技会の GLOBAL 部門で第 1 位であった．詳細は今後分析する必要があるが，order encoding の有効性が明確になったといえる.
- CSP ソルバー競技会での Sugar+minisat と Sugar+picosat の成績を比較すると，2-ARY-INT と GLOBAL 部門で Sugar+picosat のほうが上

表 8　Sugar+minisat と Sugar+picosat の比較

	Sugar+minisat			Sugar+picosat		
Category	SAT+UNSAT	SAT	UNSAT	SAT+UNSAT	SAT	UNSAT
2-ARY-EXT	470	278	192	443	280	163
2-ARY-INT	484	257	227	486	261	225
N-ARY-EXT	370	179	191	347	178	169
N-ARY-INT	486	399	87	481	393	88
GLOBAL	405	252	153	424	273	151
TOTAL	2215	1365	850	2181	1385	796

位，その他の部門では Sugar+minisat のほうが上位となった．表 8 に示すように，問題の充足可能性別に分類して比較すると，充足可能な問題では Sugar+picosat が優れ，充足不能な問題では Sugar+minisat が優れていた．これは，頻繁なリスタートにより充足可能な SAT 問題での性能向上を実現した PicoSAT の効果によるものといえる．

- Max-CSP ソルバー競技会では Sugar が Sugar++よりも優れた結果だった．Sugar++は，MiniSat のインクリメンタル探索機能を利用することにより，効率的な求解を目指している．しかし，そのため MiniSat のプロセスを終了させることなく継続して動作させており，MiniSat のメモリ使用量が増大する．それが原因となり競技会の実行環境では，メモリオーバーが生じていた．メモリ消費量を抑えながらインクリメンタル探索を実現する方法の検討が必要である．

7 おわりに

本論文では，整数有限領域上の線形の制約充足問題および最大制約充足問題を対象として，order encoding による SAT 変換方法，およびそれを実現した制約ソルバー Sugar について述べた．

Order encoding を用いた場合，SAT ソルバーでの基本動作である単位伝播が，整数変数のドメインに対する範囲伝播に対応し，direct encoding 等の他の SAT 変換方法と比較して高速な推論を実現できる．また，SAT ソルバーにおける矛盾節の学習による探索空間の枝刈り等の探索技法と組み合わさり，既存の制約ソルバーとは異なった特徴を持つ探索が実現されている．

また，第 3 回国際 CSP ソルバー競技会および Max-CSP ソルバー競技会での実行結果を元に性能評価について述べた．CSP ソルバー競技会で Sugar は GLOBAL 部門で第 1 位となり，Max-CSP ソルバー競技会で 2-ARY-INT, N-ARY-INT, GLOBAL の 3 部門で第 1 位であった．

なお，2009 年に開催された第 4 回国際 CSP ソルバー競技会で，Sugar は 3 部門で再び第 1 位となった．これらの 3 部門は GLOBAL 部門が 3 つに分かれたものである．Max-CSP ソルバー競技会は，Sugar のみの参加だったため開催されなかった．

以上のように，order encoding による SAT 変換方法は整数上の制約充足問題の解決に非常に有効な方法であり，今後，様々な分野への応用が期待される．

謝辞

本研究の一部は科学研究費補助金 基盤研究 (A)「制約最適化問題の SAT 変換による解法とその並列分散処理に関する研究」(課題番号 20240003) の助成を受けたものである．有益な助言をいただいた共同研究者の方々に感謝いたします．

また，研究にご協力いただいた大西秀志君，北川哲君，塩出雅史君，志賀彰君，中川雅也君，多賀明子さん，田島宏史君に感謝いたします．

参 考 文 献

[1] 番原睦則, 田村直之: SAT によるシステム検証, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 122–129.
[2] Biere, A.: PicoSAT Essentials, *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 4(2008), pp. 75–97.
[3] Biere, A., Heule, M., van Maaren, H. and Walsh, T.(eds.): *Handbook of Satisfiability*, IOS Press, 2009.

[4] Bordeaux, L., Hamadi, Y. and Zhang, L.: Propositional Satisfiability and Constraint Programming: a Comparative Survey, *ACM Computing Surveys*, Vol. 38, No. 4(2006).
[5] Crawford, J. M. and Baker, A. B.: Experimental Results on the Application of Satisfiability Algorithms to Scheduling Problems, in *Proceedings of the 12th National Conference on Artificial Intelligence* (*AAAI-94*), 1994, pp. 1092–1097.
[6] de Kleer, J.: A Comparison of ATMS and CSP Techniques, in *Proceedings of the 11th International Joint Conference on Artificial Intelligence* (*IJCAI 89*), 1989, pp. 290–296.
[7] Eén, N. and Sörensson, N.: An Extensible SAT-solver, in *Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing* (*SAT 2003*), 2003, pp. 502–518.
[8] Gent, I. P.: Arc Consistency in SAT, in *Proceedings of the 15th European Conference on Artificial Intelligence* (*ECAI 2002*), 2002, pp. 121–125.
[9] Inoue, K., Soh, T., Ueda, S., Sasaura, Y., Banbara, M. and Tamura, N.: A Competitive and Cooperative Approach to Propositional Satisfiability, *Discrete Applied Mathematics*, Vol. 154, No. 16(2006), pp. 2291–2306.
[10] 井上克巳, 田村直之: SAT ソルバーの基礎, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 57–67.
[11] 井上克巳, 田村直之 (編): 特集「最近の *SAT* 技術の発展」, 人工知能学会誌, Vol. 25, No. 1 (2010).
[12] Iwama, K. and Miyazaki, S.: SAT-Variable Complexity of Hard Combinatorial Problems, in *Proceedings of the IFIP 13th World Computer Congress*, 1994, pp. 253–258.
[13] Kasif, S.: On the Parallel Complexity of Discrete Relaxation in Constraint Satisfaction Networks, *Artificial Intelligence*, Vol. 45, No. 3(1990), pp. 275–286.
[14] Kautz, H. A., McAllester, D. A. and Selman, B.: Encoding Plans in Propositional Logic, in *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning* (*KR'96*), 1996, pp. 374–384.
[15] 鍋島英知: SAT によるプランニングとスケジューリング, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 114–121.
[16] 鍋島英知, 宋剛秀: 高速 SAT ソルバーの原理, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 68–76.
[17] Nabeshima, H., Soh, T., Inoue, K. and Iwanuma, K.: Lemma Reusing for SAT based Planning and Scheduling, in *Proceedings of the International Conference on Automated Planning and Scheduling 2006* (*ICAPS'06*), 2006, pp. 103–112.
[18] O'Mahony, E., Hebrard, E., Holland, A., Nugent, C. and O'Sullivan, B.: Using Case-Based Reasoning in an Algorithm Portfolio for Constraint Solving, in *Proceedings of the Second International CSP Solver Competition*, 2008, pp. 53–62.
[19] Prasad, M. R., Biere, A. and Gupta, A.: A Survey of Recent Advances in SAT-based Formal Verification, *Software Tools for Technology Transfer*, Vol. 7, No. 2(2005), pp. 156–173.
[20] Prestwich, S.: *Handbook of Satisfiability*, IOS Press, 2009, chapter 12: CNF Encodings, pp. 75–97.
[21] Sanchez, M., Bouveret, S., de Givry, S., Heras, F., Jégou, P., Larrosa, J., Ndiaye, S., Rollon, E., Schiex, T., Terrioux, C., Verfaillie, G. and Zytnicki, M.: Max-CSP Competition 2008: toulbar2 Solver Description, in *Proceedings of the Second International CSP Solver Competition*, 2008, pp. 63–70.
[22] 多賀明子, 田村直之, 北川哲, 番原睦則: グリッド計算環境上でのショップ・スケジューリング問題の SAT 変換による解法, スケジューリング・シンポジウム 2007 講演論文集, 2007, pp. 109–114.
[23] 田島宏史: SAT 変換に基づく制約ソルバーの高速化に関する研究, 修士論文, 神戸大学大学院自然科学研究科, 2006.
[24] Tamura, N. and Banbara, M.: Sugar: a CSP to SAT Translator Based on Order Encoding, in *Proceedings of the Second International CSP Solver Competition*, 2008, pp. 65–69.
[25] 田村直之, 多賀明子, 番原睦則, 宋剛秀, 鍋島英知, 井上克巳: ショップ・スケジューリング問題の SAT 変換による解法, スケジューリング・シンポジウム 2007 講演論文集, 2007, pp. 97–102.
[26] Tamura, N., Taga, A., Kitagawa, S. and Banbara, M.: Compiling Finite Linear CSP into SAT, *Constraints*, Vol. 14, No. 2(2009), pp. 254–272.
[27] Tamura, N., Tanjo, T. and Banbara, M.: System Description of a SAT-based CSP Solver Sugar, in *Proceedings of the Third International CSP Solver Competition*, 2008, pp. 71–75.
[28] 田村直之, 丹生智也, 番原睦則: 制約最適化問題と SAT 符号化, 人工知能学会誌, Vol. 25, No. 1(2010), pp. 77–85.
[29] Tanjo, T., Tamura, N. and Banbara, M.: Sugar++: a SAT-based MAX-CSP/COP Solver, in *Proceedings of the Third International CSP Solver Competition*, 2008, pp. 77–82.
[30] van Dongen, M., Lecoutre, C. and Roussel, O.(eds.): *Proceedings of the Second International CSP Solver Competition*, 2006.
[31] van Hoeve, W.-J.: The Alldifferent Constraint: A Survey, in *Proceedings of the 6th Annual Workshop of the ERCIM Working Group on Constraints*, 2001.
[32] Walsh, T.: SAT v CSP, in *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming* (*CP 2000*), 2000, pp. 441–456.

A 3×3 魔方陣の Sugar による SAT 変換例

ここでは，図 5 に示した制約充足問題の Sugar での変換過程を説明する．なお，問題中の整数変数 `x1`, `x2` 等は x_1, x_2 等と表記する．

A.1 前処理

まず前処理段階で，alldifferent 制約は 5.3 節の方法に従い，以下の制約に変換される．

$$x_i \neq x_j \qquad (1 \leq i < j \leq 9)$$
$$\neg(x_1 \leq 8) \vee \cdots \vee \neg(x_9 \leq 8)$$
$$x_1 \leq 1 \vee \cdots \vee x_9 \leq 1$$

下の 2 つは鳩の巣原理により導入された制約である．

さらに制約 $x_i \neq x_j$ は，図 3 に従い一旦 $(x_i - x_j \leq -1) \vee (x_j - x_i \leq -1)$ の形を経て，Tseitin 変換により以下の制約に変換される (q_{ij}, q_{ji} は新しいブール変数)．

$$q_{ij} \vee q_{ji}$$
$$\neg q_{ij} \vee x_i - x_j \leq -1$$
$$\neg q_{ji} \vee x_j - x_i \leq -1$$

制約 $x_i + x_j + x_k = 15$ も以下に変換される．

$$x_i + x_j + x_k \leq 15$$
$$-x_i - x_j - x_k \leq -15$$

最終的に前処理後の制約は以下のようになる．

- 各 $1 \leq i < j \leq 9$ について
$$q_{ij} \vee q_{ji}$$
$$\neg q_{ij} \vee x_i - x_j \leq -1$$
$$\neg q_{ji} \vee x_j - x_i \leq -1$$
- 各 $(i,j,k) \in \{(1,2,3),(4,5,6),(7,8,9),(1,4,7),(2,5,8),(3,6,9),(1,5,9),(3,5,7)\}$ について
$$x_i + x_j + x_k \leq 15$$
$$-x_i - x_j - x_k \leq -15$$

A.2 Order encoding による変換

次に，各整数変数および各制約は order encoding により SAT 問題の節に変換される．以下では，見やすさのため命題変数 p_{xa} を $p(x \leq a)$ と表記する．

各整数変数 $x_i \in \{1, 2, \ldots, 9\}$ は，以下の 7 個の節に変換される．

$$\neg p(x_i \leq 1) \vee p(x_i \leq 2)$$
$$\cdots$$
$$\neg p(x_i \leq 7) \vee p(x_i \leq 8)$$

鳩の巣原理による 2 つの制約は，以下の 2 個の節に変換される．

$$\neg p(x_1 \leq 8) \vee \cdots \vee \neg p(x_9 \leq 8)$$
$$p(x_1 \leq 1) \vee \cdots \vee p(x_9 \leq 1)$$

各制約 $q_{ij} \vee q_{ji}$ はそのままで節である．

各制約 $\neg q_{ij} \vee x_i - x_j \leq -1$ は，以下の 9 個の節に変換される．

$$\neg q_{ij} \vee \neg p(x_j \leq 1)$$
$$\neg q_{ij} \vee p(x_i \leq 1) \vee \neg p(x_j \leq 2)$$
$$\cdots$$
$$\neg q_{ij} \vee p(x_i \leq 7) \vee \neg p(x_j \leq 8)$$
$$\neg q_{ij} \vee p(x_i \leq 8)$$

各制約 $\neg q_{ji} \vee x_j - x_i \leq -1$ も同様に変換される．

各制約 $x_i + x_j + x_k \leq 15$ は，以下の 60 個の節に変換される．

$$p(x_j \leq 5) \vee p(x_k \leq 8)$$
$$\cdots$$
$$p(x_i \leq 1) \vee p(x_j \leq 4) \vee p(x_k \leq 8)$$
$$\cdots$$
$$p(x_i \leq 8) \vee p(x_j \leq 5)$$

各制約 $-x_i - x_j - x_k \leq -15$ も，同様に以下の 60 個の節に変換される．

$$\neg p(x_i \leq 1) \vee \neg p(x_j \leq 4)$$
$$\neg p(x_i \leq 1) \vee \neg p(x_j \leq 5) \vee \neg p(x_k \leq 8)$$
$$\neg p(x_i \leq 1) \vee \neg p(x_j \leq 6) \vee \neg p(x_k \leq 7)$$
$$\cdots$$
$$\neg p(x_j \leq 4) \vee \neg p(x_k \leq 1)$$

最終的に元の制約充足問題は，144 個のブール変数および 1709 個の節からなる SAT 問題に変換される．

この SAT 問題を MiniSat で解いた場合，数ミリ秒で充足可能と判定され解が得られる (Xeon 2.80GHz, メモリ 4GB の計算機を使用)．

この解は，Sugar により元の制約充足問題の解に逆変換され，結果として出力される．