人工知能学会研究会資料
SIG-FPAI-B404-12

制約充足問題のハイブリッド符号化に向けて

Towards a Hybrid Encoding of Constraint Satisfaction Problems

宋剛秀 [1]　佐古田淳史 [2]　番原睦則 [1]　田村直之 [1]
Takehide Soh[1]　Atsushi Sakoda[2]　Mutsunori Banbara[1]　Naoyuki Tamura[1]

[1] 神戸大学情報基盤センター
[1] Information Science and Technology Center, Kobe University
[2] 神戸大学大学院システム情報学研究科
[2] Graduate School of System Informatics, Kobe University

Abstract: In this paper, we propose a hybrid encoding of Constraint Satisfaction Problems (CSPs) to propositional logic (especially SAT and PB). There have been proposed several encodings of CSPs to propositional logic, such as direct encoding, order encoding, and binary encoding. However, each of these encodings has its pros and cons. For example, the order encoding which is used in a SAT-based constraint solver Sugar, showed a good performance on many problem instances, but it can not be applied to CSP instances with very large domain size. On the other hand, the binary encoding is applicable to those large instances, but the generated SAT instances take more time to be solved than the order encoded instances in general. The hybrid encoding proposed in this paper mixes the order and binary encodings so that it can be applied to very large instances while it keeps the good performance of the order encoding for small instances. We evaluated the performance of the proposed hybrid encoding with our implementation named Hysca. In our evaluation using 1453 instances of the 2009 CSP solver competition, Hysca succeeded to solve 1075 instances, which is 56 more than the order encoding, and 53 more than the binary encoding.

1 はじめに

制約充足問題 (CSP; Constraint Satisfaction Problem) は，与えられた制約を満たす解を探索する問題であり，人工知能などの分野における多くの組合せ問題は，CSP として定式化できることが知られている [18].

一方，与えられた命題論理の充足可能性判定問題 (SAT) や，その拡張である擬似ブール制約充足問題 (PB) について，それらを解くプログラムである SAT ソルバーや PB ソルバーの性能が大きく向上している [4, 11, 17]．それを背景として，CSP を SAT/PB に変換 (符号化) し，高速な SAT/PB ソルバーを用いて解く方法が成功を収めている [23, 25, 20].

CSP の命題論理への符号化としては，順序符号化 [5, 24]，2 進符号化 [12, 9] など，様々な方法が提案されている．しかし，いずれの符号化も長所と短所が存在する．特に順序符号化は，2008 年および 2009 年の国際 CSP ソルバー競技会 [13] の複数部門で優勝した SAT 型制約ソルバー Sugar [*1] に採用されており，多くの問題に対して優れた性能を示している．しかし，順序符号化は変数のドメインサイズが大きい場合，生成される SAT 問題のサイズが巨大になり，解くことができないという問題点がある．一方，2 進符号化は，順序符号化に性能的には劣るが，大きな問題にも対応できるという利点を持つ．

したがって，複数の符号化を融合させた **ハイブリッド符号化** の研究が重要になる．特に，直接符号化 [6, 28] と順序符号化のハイブリッドについては論文 [10], BEE [16], meSAT [21] で研究されている．これらはいずれも各変数を直接符号化と順序符号化の両方で符号化し，さらにそれらの符号化を関連付けるためチャネリング制約と呼ばれるものを追加している．この方法 (ここではチャネリング方式と呼ぶ) は，直接符号化と順序符号化のハイブリッドについては成功したが，2 進符号化との

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

ハイブリッドには利用できない．なぜなら，2 進符号化した変数を，直接符号化あるいは順序符号化と関連付けた場合，チャネリング制約が膨大になり，2 進符号化の利点が失われてしまうからである．

そこで本論文では，チャネリング制約が不要な新しいハイブリッド符号化の方法を提案し，それを順序符号化と 2 進符号化のハイブリッドに適用する．また，提案した方法を実装した Hysca を用い，ハイブリッド符号化の性能を評価する [19]．

以下，2 節で CSP および 0-1 CSP, PB, SAT を定義する．3 節では，既存の符号化を統一的に説明する枠組みとして標準的変換を提案する．4 節では，標準的変換の概念を用いて既存の符号化を定義する．5 節で，標準的変換の概念を用い，順序符号化と 2 進符号化を融合させたハイブリッド符号化を提案する．チャネリング制約が不要なハイブリッドを実現できている点が特徴である．6 節では，提案したハイブリッド符号化を実装した Hysca を説明する．7 節では，国際 CSP ソルバー競技会のベンチマーク問題を用いて，ハイブリッド符号化を既存の順序符号化および 2 進符号化と比較・評価する．最後に，8 節で結論を述べる．

2　制約充足問題

本節では制約充足問題 (CSP; Constraint Satisfaction Problem) を定義し，さらにその一種として 0-1 制約充足問題 (0-1 CSP)，擬似ブール制約充足問題 (PB)，命題論理の充足可能性判定問題 (SAT) を定義する．

2.1　CSP

一般の制約充足問題では，整数，実数，集合等のドメイン上の様々な制約を対象としているが，本稿では整数有限領域上の線形の制約を対象とする．なお，実用的な制約充足問題の多くは，整数有限領域上の線形制約で表現できることが知られている．

$\mathcal{X}$ を **変数** の集合とする (以下 x, y, x_1, y_1 等で変数を表す)．$x_1, x_2, \ldots, x_n$ が変数，$a_1, a_2, \ldots, a_n$ および c が整数定数の時，$\sum_{i=1}^{n} a_i x_i$ の形の式を **線形和** (linear summation) と呼び，$\sum_{i=1}^{n} a_i x_i \geq c$ の形の式を **線形比較** (linear comparison) と呼ぶ (以下 S, S_1 等で線形和を表し，E, E_1 等で線形比較を表す)．ただし $a_i \neq 0$ かつ x_i はすべて異なると仮定する．線形比較 $S \geq c$ の否定 $\neg(S \geq c)$ を $-S \geq -c+1$ で定義する．なお，線形比較の比較演算子が $=, \neq, \leq$ 等の場合も簡単に $\geq$ の線形比較に書き換えることができるため，比較演算子を $\geq$ に限定しても一般性を失わない．

線形比較の有限集合を **制約** (constraint) と呼び，制約の有限集合を **制約集合** (constraint set) と呼ぶ．以下 C, C_1 等で制約を表し，$\boldsymbol{C}$ 等の斜体太字で制約集合を表す．

定義 1．制約充足問題 (**CSP**; Constraint Satisfaction Problem) は，以下を満たす組 $(X, Dom, \boldsymbol{C})$ である．

- X は **変数** の有限集合 ($X \subset \mathcal{X}$)．
- Dom は各変数の取りうる値の集合である **ドメイン** を定める関数．ここでは，ドメインは整数の有界の範囲とする．
- $\boldsymbol{C}$ は X 上の線形比較から成る制約集合．

制約集合 $\boldsymbol{C} = \{C_1, \ldots, C_n\}$ は制約 C_i の連言を意味し，$C_1 \wedge \cdots \wedge C_n$ あるいは C_i の列で表すことがある．また，制約 $C = \{E_1, \ldots, E_m\}$ は線形比較 E_j の選言を意味し，$E_1 \vee \cdots \vee E_m$ で表すことがある．すなわち制約集合は，線形比較の CNF 式 (Conjunctive Normal Form 式) である．制約集合間の演算 $\wedge, \vee$ を以下のように定義する．

$$(\boldsymbol{C}_1 \wedge \boldsymbol{C}_2) = (\boldsymbol{C}_1 \cup \boldsymbol{C}_2)$$
$$(\boldsymbol{C}_1 \vee \boldsymbol{C}_2) = \{C_1 \cup C_2 \mid C_1 \in \boldsymbol{C}_1,\ C_2 \in \boldsymbol{C}_2\}$$

線形比較 E 中に現れる変数の集合を $Var(E)$ で表す．制約 C についても同様に $Var(C)$ で表示する．変数 x のドメイン $Dom(x)$ の下限と上限を，それぞれ $lb(x)$ と $ub(x)$ で表す．同様に線形和 S の下限と上限を，それぞれ $lb(S)$ と $ub(S)$ で表す．

代入 (substitution) は変数から線形和への写像であり，変数 x と線形和 S との組 $x \mapsto S$ の集合で表す．E が線形比較で σ が代入の時，$E\sigma$ で E 中の変数 x の各出現を線形和 $\sigma(x)$ で置き換えた結果を表す．制約 C や他の式についても同様に $C\sigma$ 等で代入の結果を表す．

CSP $(X, Dom, \boldsymbol{C})$ において，X から $\mathbb{Z}$ への関数 α がすべての変数 $x \in X$ について $\alpha(x) \in Dom(x)$ を満たす時 α を **値割当て** (assignment) と呼ぶ．値割当て α の定義域を以下のように線形比較，制約および制約集合に拡張する．

$$\alpha(\textstyle\sum a_i x_i \geq c) = \begin{cases} 1 & (\sum a_i \alpha(x_i) \geq c \text{ の時}) \\ 0 & (\text{otherwise}) \end{cases}$$
$$\alpha(\{E_1, \ldots, E_m\}) = \begin{cases} 1 & (\bigvee_{j=1}^{m} \alpha(E_j) = 1 \text{ の時}) \\ 0 & (\text{otherwise}) \end{cases}$$
$$\alpha(\{C_1, \ldots, C_n\}) = \begin{cases} 1 & (\bigwedge_{i=1}^{n} \alpha(C_i) = 1 \text{ の時}) \\ 0 & (\text{otherwise}) \end{cases}$$

線形比較，制約，制約集合に対する値をその真理値と呼ぶ．値 1 と 0 はそれぞれ **真** と **偽** を表す．

CSP $(X, Dom, \boldsymbol{C})$ において，値割当て α が $\boldsymbol{C}$ 中のすべての制約 C に対し $\alpha(C) = 1$ の時，α はその CSP

を **充足する** といい，α を **解** という．CSP を充足する値割当てが存在する時，その CSP は **充足可能** (SAT; satisfiable) であるといい，充足可能でない時 **充足不能** (UNSAT; unsatisfiable) であるという．

2.2 0-1 CSP, PB, SAT

CSP $(X, Dom, \boldsymbol{C})$ において，$Dom(x) \subset \{0,1\}$ の場合，x を **ブール変数** と呼ぶ (以下 p, q, p_1, q_1 等でブール変数を表す)．

1 つのブール変数から成る線形比較を **リテラル** (literal) と呼ぶ (以下 L, L_1 等でリテラルを表す)，リテラルでない線形比較を **複雑な線形比較** と呼ぶ．リテラルの真理値は 1, 0, p, $1-p$ のいずれかに等しい．リテラルとその真理値表現を同一視することがある．

定義 2. 複雑な線形比較 (リテラルでない線形比較) を高々 1 つ含む制約を**単純な制約** (simple constraint) と呼び，すべての制約が単純な CSP を **単純 CSP** と呼ぶ．

$\{p, \neg q\}$, $\{p_1, p_2, x - y \geq 0\}$ は単純な制約の例である．

定義 3. すべての変数がブール変数の制約を **0-1 制約** と呼び，すべての制約が 0-1 制約の CSP を **0-1 CSP** と呼ぶ．

$\{p_1 - p_2 \geq 0, p_2 - p_3 \geq 0\}$ は 0-1 制約の例である．

定義 4. 1 つの線形比較からなる 0-1 制約を **擬似ブール制約** (あるいは **PB 制約**) と呼び，すべての制約が擬似ブールの CSP を **擬似ブール制約充足問題** (**PB**) と呼ぶ．

$\{p_1 + 2p_2 \geq 1\}$ は擬似ブール制約の例である．擬似ブール制約は単純 0-1 制約，PB は単純 0-1 CSP である．

定義 5. すべての線形比較がリテラルの制約を **節** (clause) と呼び，すべての制約が節の CSP を **命題論理の充足可能性判定問題** (**SAT**) と呼ぶ．

$\{p, \neg q\}$ は節の例である．節は単純 0-1 制約，SAT は単純 0-1 CSP である．

3 標準的変換

3.1 充足同値な変換

CSP から CSP への **変換** (あるいは **符号化**) τ において，元の CSP P と変換後の CSP P' の充足可能性が一致する時，変換 τ は **充足同値** (equi-satisfiable) であるという．変換 τ_1, τ_2 が充足同値な時，合成 $\tau_1 \circ \tau_2$ [*2] もまた充足同値な変換である．

[*2] $((f \circ g)(x) = g(f(x)))$

充足同値な変換として，いくつかの種類が考えられる．

1. **制約置換による変換** (Translation by constraint substitution): 補助ブール変数を導入し，線形比較を補助ブール変数に書き換える方法である．Tseitin 変換はこの変換の例である．
2. **項置換による変換** (Translation by term substitution): 補助変数を導入し，部分和を補助変数に書き換える方法である．例えば $4x_1 + 3x_2 + 2x_3 + x_4 \geq 5$ に対し，補助変数 t を導入し $4x_1 + 3x_2 + t \geq 5$, $t = 2x_3 + x_4$ に変換する．
3. **変数置換による変換** (Translation by variable substitution): 変数をそれと同等の式に置き換える方法である．例えば $Dom(x) = \{0,1,2,3\}$ の時，x の各ビットを表すブール変数 x_0, x_1 を導入し，x を $2x_1 + x_0$ で置き換える．後述の 1 進符号化，2 進符号化はこの変換の例である．
4. **値具体化による変換** (Translation by value instantiation): 制約中に現れる変数や式の値を場合分けし，元の制約をそれらの連言で表す方法である．後述の順序符号化はこの変換の例である．

以下では，これらを統一的な枠組みで説明する．

3.2 標準的変換の充足同値な条件

上で述べた充足同値な変換を，統一的な枠組みで定義するため，変換 τ を以下の組 $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める．ただし $(X, Dom, \boldsymbol{C})$ を元の CSP とする．

- U は追加する変数の集合を表す． $X \cap U = \emptyset$ とする．
- V は削除する変数の集合を表す． $V \subset X$ とする．
- D は追加する変数のドメインを定める関数である．
- $\boldsymbol{A}$ は追加する制約集合を表す．
- $\|\cdot\|$ は与えられた線形比較を制約集合に変換する関数である．

この時 τ を $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定められる **標準的変換** (standard translation) と呼び，変換後の CSP $(X', Dom', \boldsymbol{C}')$ を以下で定義する．

$$X' = (X \setminus V) \cup U$$

$$Dom'(x) = \begin{cases} Dom(x) & (x \in X \setminus V) \\ D(x) & (x' \in U) \end{cases}$$

$$\boldsymbol{C}' = \boldsymbol{A} \cup \{C' \mid C \in \boldsymbol{C},\ C' \in \textstyle\bigvee_{E \in C} \|E\|\}$$

以下の命題は，標準的変換が充足同値になる十分条件を与える．

命題 1. CSP $P = (X, Dom, \boldsymbol{C})$ から CSP $P' = (X', Dom', \boldsymbol{C}')$ への標準的変換 τ が組 $(U, V, D, \boldsymbol{A}, \|\cdot\|)$

で定められ，以下を満たす時，τ は充足同値である．

(1) P の任意の値割当て α に対し，P' の値割当て α' が存在し，任意の線形比較 $E \in \boldsymbol{C}$ に対し $\alpha(E) \le \alpha'(\|E\|)$ かつ $\alpha'(\boldsymbol{A}) = 1$ である．

(2) P' の任意の値割当て α' に対し，P の値割当て α が存在し，任意の線形比較 $E \in \boldsymbol{C}$ に対し $\alpha(E) \ge \alpha'(\|E\|)$ である．

Proof. まず P が充足可能ならば P' が充足可能を示す．P の解を α とすると，(1) の α' が存在する．$C = \{E_1, \ldots, E_m\}$ を P の任意の制約とすると，ある E_j について $\alpha(E_j) = 1$ である．(1) より $\alpha'(\|E_j\|) = 1$ が導かれ，制約集合 $\|E_j\|$ 中のすべての制約 C'_j について $\alpha(C'_j) = 1$ である．したがって $C' \in \|C\|$ について，$C' \subset C'_j$ より $\alpha(C') = 1$ である．さらに (1) の $\alpha'(\boldsymbol{A}) = 1$ より α' は P' の解であることが示される．

次に P' が充足可能ならば P が充足可能を示す．P' の解を α' とすると，(2) の α が存在する．ある $C = \{E_1, \ldots, E_m\} \in \boldsymbol{C}$ について $\alpha(C) = 0$ と仮定し，矛盾を導く．$\alpha(C) = 0$ より，すべての E_j について $\alpha(E_j) = 0$ で，(2) より $\alpha'(\|E_j\|) = 0$ である．すなわち，すべての E_j に対しある $C'_j \in \|E_j\|$ が存在し，$\alpha'(C'_j) = 0$ であり，それらの C'_j から成る $C' = \bigcup_{j=1}^{m} C'_j$ を考えると $\alpha'(C') = 0$ である．これは α' が P' の解であることに矛盾する． □

後述の Tseitin 変換 τ^{t}, 部分和置換 τ^{s}, 1 進符号化 τ^{u}, 2 進符号化 τ^{b}, 順序符号化 τ^{o} はすべて充足同値な標準的変換である．

4 既知の充足同値な変換

本節では，部分項置換と Tseitin 変換を標準的変換として定義し，さらに既知の **SAT 符号化** (CSP から SAT への充足同値かつ復号可能な符号化) のうち 1 進符号化，2 進符号化 [12, 9]，順序符号化 [5, 24] を標準的変換として定義する．直接符号化 [6, 28]，コンパクト順序符号化 [26, 27] も，同様に定義できるが本稿の対象としない．また，最後の単純 0-1 CSP から PB への変換，PB から SAT への変換について述べる．

4.1 Tseitin 変換

CSP $P = (X, Dom, \boldsymbol{C})$ 中の線形比較 E を新しいブール変数で置き換える変換を Tseitin 変換と呼ぶ．Tseitin 変換 τ^{t}_E を標準的変換として以下の $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める (t は Tseitin を表す)．ただし p_E は新しいブール変数である．

$$
\begin{aligned}
U &= \{p_E\} \\
V &= \emptyset \\
D(p) &= \{0, 1\} \\
\boldsymbol{A} &= \{\{\neg p_E, E\}\} \\
\|F\| &= \begin{cases} \{\{p_E\}\} & (F = E) \\ \{\{F\}\} & (F \ne E) \end{cases}
\end{aligned}
$$

命題 2. Tseitin 変換 τ^{t}_E は充足同値である．

Proof. τ^{t}_E が命題 1 の条件を満たすことを示す．

(1) α を P の値割当てとする．α' を $\alpha'(x) = \alpha(x)$ ($x \in X \setminus V$ の時), $\alpha'(p_E) = \alpha(E)$ で定義する．この時 P 中の任意の線形比較 F について $\alpha(F) \le \alpha'(\|F\|)$ を示す．$F = E$ の時 $\|F\| = \{\{p_E\}\}$, $F \ne E$ の時 $\|F\| = \{\{F\}\}$ で，どちらについても $\alpha(E) = \alpha'(\|E\|)$ が成り立つ．また $\alpha'(\{\{\neg p_E, E\}\}) = 1$ である．

(2) α' を P' の値割当てとする．α は p_E を除き α' と同一と定義する．(1) と同様にして P 中の任意の線形比較 F について $\alpha(F) \ge \alpha'(\|F\|)$ が示される． □

単純でない制約 C 中の複雑な線形比較 E のすべてに対し Tseitin 変換 τ^{t}_E を行うことで，CSP から単純 CSP への充足同値な変換 τ^{t} が実現できる．

4.2 部分項置換

CSP $P = (X, Dom, \boldsymbol{C})$ 中の部分和 S を新しい変数で置き換える変換を部分項置換と呼ぶ．部分項置換 τ^{s}_S を標準的変換として以下の $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める (s は subterm を表す)．ただし x_S は新しい変数である．

$$
\begin{aligned}
U &= \{x_S\} \\
V &= \emptyset \\
D(x_S) &= \{lb(S) \mathinner{..} ub(S)\} \\
\boldsymbol{A} &= \{\{S - x_S \ge 0\}\} \\
\|E\| &= \begin{cases} \{\{x_S + S' \ge c\}\} & (E = (S + S' \ge c)) \\ \{\{E\}\} & (\text{otherwise}) \end{cases}
\end{aligned}
$$

命題 3. 部分項置換 τ^{s}_S は充足同値である．

Proof. τ^{s}_S が命題 1 の条件を満たすことを示す．

(1) α を P の値割当てとする．α' を $\alpha'(x) = \alpha(x)$ ($x \in X \setminus V$ の時), $\alpha'(x_S) = \alpha(S)$ で定義し，P 中の任意の線形比較 E について $\alpha(E) \le \alpha'(\|E\|)$ を示す．$E = (S + S' \ge c)$ の時 $\alpha'(\|E\|) = \alpha'(x_S + S' \ge c) = \alpha'(S + S' \ge c) = \alpha(S + S' \ge c)$ であり，その他の時 $\|E\| = \{\{E\}\}$ で，どちらについても $\alpha(E) = \alpha'(\|E\|)$ が成り立つ．また $\alpha'(\boldsymbol{A}) = 1$ である．

(2) α' を P' の値割当てとする．α は x_S を除き α' と同一と定義する．(1) と同様にして P 中の任意の線形比較 E について $\alpha(E) \ge \alpha'(\|E\|)$ が示される． □

4.3 1進符号化

1進符号化 (unary encoding) は，CSP から 0-1 CSP への充足同値な変換である．元の CSP の変数 x に対し，$ub(x) - lb(x)$ 個の新しいブール変数 $p_{x\#d}$ を導入し，x をそれらの和で表す．

例えば $Dom(x) = \{1..5\}$ なる変数 x の符号化を考える．

- 以下の 4 つのブール変数を導入する．

$$p_{x\#2} \quad p_{x\#3} \quad p_{x\#4} \quad p_{x\#5}$$

- 元の制約中のすべての x を以下の式で置き換える．

$$1 + p_{x\#2} + p_{x\#3} + p_{x\#4} + p_{x\#5}$$

変数 x に対する 1 進符号化 τ_x^{u} を標準的変換として以下の $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める (u は unary を表す)．

$$\begin{aligned} U &= \{p_{x\#d} \mid lb(x) < d \le ub(x)\} \\ V &= \{x\} \\ D(p) &= \{0,1\} \quad (p \in U) \\ \boldsymbol{A} &= \emptyset \\ \|E\| &= \{\{E\sigma\}\} \\ \sigma &= \{x \mapsto lb(x) + \textstyle\sum_{d=lb(x)+1}^{ub(x)} p_{x\#d}\} \end{aligned}$$

なお x がブール変数の時 τ_x^{u} により得られる CSP は，変数名の変更を除いて元の CSP と同等である．

命題 4. 1 進符号化 τ_x^{u} は充足同値である．

Proof. τ_x^{u} が命題 1 の条件を満たすことを示せば良い．詳細は省略する． □

4.4 2進符号化

2進符号化 (binary encoding) は，CSP から 0-1 CSP への充足同値な変換である．元の CSP の変数 x に対し，$x - lb(x)$ の 2 進表現の i ビット目を表す新しいブール変数 $p_{x,i}$ を導入し，x をそれらの重み付き和で表す [12, 9]．

例えば $Dom(x) = \{1..5\}$ なる変数 x の符号化を考える．

- 以下の 3 つのブール変数を導入する．

$$p_{x,0} \quad p_{x,1} \quad p_{x,2}$$

  $p_{x,i}$ は $x - lb(x)$ の i ビット目を表す (LSB を 0 ビット目とする)．
- 導入したブール変数に対し，以下の制約を導入する．

$$\{1 + 2^0 p_{x,0} + 2^1 p_{x,1} + 2^2 p_{x,2} \le 5\}$$

- 元の制約中のすべての x を以下の式で置き換える．

$$1 + 2^0 p_{x,0} + 2^1 p_{x,1} + 2^2 p_{x,2}$$

変数 x に対する 2 進符号化 τ_x^{b} を標準的変換として以下の $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める (b は binary を表す)．ただし $m_x = \lceil \log_2(ub(x) - lb(x)) \rceil$ とする．

$$\begin{aligned} U &= \{p_{x,i} \mid 0 \le i \le m_x\} \\ V &= \{x\} \\ D(p) &= \{0,1\} \quad (p \in U) \\ \boldsymbol{A} &= \begin{cases} \emptyset & (ub(x) = 2^{m_x}) \\ \{x\sigma \le ub(x)\} & (\text{otherwise}) \end{cases} \\ \|E\| &= \{\{E\sigma\}\} \\ \sigma &= \{x \mapsto lb(x) + \textstyle\sum_{i=0}^{m_x} 2^i p_{x,i}\} \end{aligned}$$

なお x がブール変数の時 τ_x^{b} により得られる CSP は，変数名の変更を除いて元の CSP と同等である．

命題 5. 2 進符号化 τ_x^{b} は充足同値である．

Proof. τ_x^{b} が命題 1 の条件を満たすことを示せば良い．詳細は省略する． □

4.5 順序符号化

順序符号化 (order encoding) は，CSP から 0-1 CSP への充足同値な変換である．元の CSP の変数 x および値 $d \in Dom(x) \setminus \{lb(x)\}$ に対し $x \ge d$ を意味する新しいブール変数 $p_{x \ge d}$ を導入する [5, 24] (引用論文では $x \ge d$ ではなく $x \le d$ にブール変数を対応させている)．

例えば $Dom(x) = Dom(y) = \{1..5\}$ なる変数 x, y および制約 $x - y \ge 0$ の符号化を考える．

- 変数 x について以下の 4 つのブール変数を導入する ($p_{x\ge1}$ は $x \ge 1$ が常に真のため不要)．変数 y についても同様とする．

$$p_{x\ge2} \quad p_{x\ge3} \quad p_{x\ge4} \quad p_{x\ge5}$$

- 導入したブール変数に対し，以下の制約を導入する．変数 y についても同様とする．

$$\{p_{x\ge2}, \neg p_{x\ge3}\}, \quad \{p_{x\ge3}, \neg p_{x\ge4}\}, \quad \{p_{x\ge4}, \neg p_{x\ge5}\}$$

  $x = 1$ は $p_{x\ge d}$ のすべてが偽の時に対応し，$x = 5$ はすべてが真の時に対応する．
- この時 $x - y \ge 0$ は，以下のように符号化できる．

$$\{\{p_{x\ge d}, \neg p_{y\ge d}\} \mid 1 < d \le 5\}$$

変数 x に対する順序符号化 τ_x^{o} を標準的変換として以下の $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定める (τ_x^{o} の o は order を表

す). ただし $d \le lb(x)$ の時 $p_{x\ge d}=1$, $d>ub(x)$ の時 $p_{x\ge d}=0$ とする.

$$
\begin{aligned}
U &= \{p_{x\ge d} \mid lb(x) < d \le ub(x)\} \\
V &= \{x\} \\
D(p) &= \{0,1\} \quad (p \in U) \\
\boldsymbol{A} &= \{\{p_{x\ge d}, \neg p_{x\ge d+1}\} \mid lb(x) < d < ub(x)\} \\
\|{\textstyle\sum_{i=1}^n} a_i x_i \ge c\| &=
\end{aligned}
$$

$$
\begin{cases}
\{\{\sum_{i=1}^n a_i x_i \ge c\}\} & (\forall x_i \ne x) \\
\{\{p_{x\ge\lceil c/a_1\rceil}\}\} & (n=1,\ x_1=x,\ a_1>0) \\
\{\{\neg p_{x\ge\lfloor c/a_1\rfloor+1}\}\} & (n=1,\ x_1=x,\ a_1<0) \\
\{\{p_{x\ge d+1},\ \sum_{j\ne i} a_j x_j \ge c - a_i d\} \mid d \in Dom(x)\} & \\
 & (n>1,\ x_i=x,\ a_i>0) \\
\{\{\neg p_{x\ge d},\ \sum_{j\ne i} a_j x_j \ge c - a_i d\} \mid d \in Dom(x)\} & \\
 & (n>1,\ x_i=x,\ a_i<0)
\end{cases}
$$

順序符号化の特徴のいくつかを述べる.

- ブール変数以外の全変数 x に τ_x^{o} を適用すると 0-1 CSP が得られる.
- ブール変数を含む全変数 x に τ_x^{o} を適用すると SAT が得られる.
- 1 進符号化を混在させることが可能である. すなわち x を含む各線形比較に対し $\|\cdot\|_x^{\mathrm{u}}$, $\|\cdot\|_x^{\mathrm{o}}$ のどちらを用いても良い.

なお x がブール変数で線形比較 E が x のリテラルの時, $\|E\|$ により得られる制約集合は, それぞれ $\{p_{x\ge 1}\}$ または $\{\neg p_{x\ge 1}\}$ であり, 元の E と同等である.

命題 6. 順序符号化 τ_x^{o} は充足同値である.

Proof. τ_x^{o} が命題 1 の条件を満たすことを示せば良い. 詳細は省略する. □

4.6 単純 0-1 CSP から PB への符号化

単純 0-1 CSP は Big-M 法により PB に変換できる.

例えば制約 $\{p, \sum a_i x_i \ge c\}$ (p はブール変数) は, $\sum a_i x_i$ の最小値を a とすると, $(c-a)p + \sum a_i x_i \ge c$ に変換できる. $p=1$ の時は無条件に成立し, $p=0$ の時は $\sum a_i x_i \ge c$ と同等になる.

Big-M 法による単純 0-1 CSP $(X, Dom, \boldsymbol{C})$ から PB $(X', Dom', \boldsymbol{C}')$ への変換 τ^{M} は以下のように定められる (τ^{M} の M は Big-M を表す). ただし $a = lb(\sum a_i x_i)$ とする.

$$
\begin{aligned}
X' &= X \\
Dom' &= Dom \\
\boldsymbol{C}' &= \{C' \mid C \in \boldsymbol{C},\ C' \in \|C\|\} \\
\|\{L_1,\dots,L_m\}\| &= \{\{\textstyle\sum_{i=1}^m L_i \ge 1\}\} \\
\|\{L_1,\dots,L_m, \textstyle\sum_{i=1}^n a_i x_i \ge c\}\| &=
\end{aligned}
$$

$$
\begin{cases}
\{\{(c-a)\sum_{j=1}^m L_j + \sum_{i=1}^n a_i x_i \ge c\}\} & (c>a) \\
\{\} & (c \le a)
\end{cases}
$$

命題 7. 変換 τ^{M} は充足同値である.

Proof. 省略. □

以上により, CSP は単純 CSP, 単純 0-1 CSP を経て PB に変換でき, Sat4j [3], clasp [8], bsolo [15], wbo [14], MiniSat+ [7] などの PB ソルバーを用いて求解できる.

4.7 PB から SAT への符号化

PB から SAT への符号化は様々な方法が提案されている [7, 2, 1, 22]. これらの方法を用いることで CSP を SAT へ符号化できる.

5 ハイブリッド符号化の提案

最初に述べたように, 順序符号化は多くの問題で優れた性能を示しているが, ドメインサイズが非常に大きい問題に対しては適していない. 一方, 2 進符号化は求解速度において順序符号化に劣るが, ドメインサイズが大きい問題に対しても適用可能である.

そこで, 本稿では順序符号化と 2 進符号化をハイブリッドした符号化を提案する [19]. 順序符号化と直接符号化をハイブリッドした符号化は BEE [16], meSAT [21] で実現されているが, 2 進符号化とのハイブリッドは存在していない.

提案する方法では, 元の CSP の変数を順序符号化するものと 2 進符号化するものの 2 種類に分類する. 分類には, 以下の方法が考えられる.

1. 変数 x のドメインサイズ $|Dom(x)|$ が一定値以下の場合に順序符号化する.
2. 変数 x が現れるすべての線形比較の項数 (アリティ) が一定値以下の場合に順序符号化する.
3. 変数 x が現れるすべての線形比較のドメイン積が一定値以下の場合に順序符号化する. 線形比較 $\sum_{i=1}^n a_i x_i \ge c$ の **ドメイン積** とは, $\frac{1}{d}\prod_{i=1}^n |Dom(x_i)|$ である. ただし $d = \max_i |Dom(x_i)|$ である. ドメイン積は, 線形比較を順序符号化した時の節数の概算を表す.

上記 3 種類について予備実験を行った所，3 番目のドメイン積を用いる方法が最も良かったため，ここではそれを採用する．

また順序符号化する変数と 2 進符号化する両方の変数が現れる線形比較については，順序符号化した変数を 1 進符号化した変数として取り扱うことで，ハイブリッド符号化を実現する．

変数 x に対する順序符号化 τ_x^{o} を定めている組を $(U^{\mathrm{o}}, V^{\mathrm{o}}, D^{\mathrm{o}}, \boldsymbol{A}^{\mathrm{o}}, \|\cdot\|^{\mathrm{o}})$, 1 進符号化 τ_x^{b} を定めている組を $(U^{\mathrm{u}}, V^{\mathrm{u}}, D^{\mathrm{u}}, \boldsymbol{A}^{\mathrm{u}}, \|\cdot\|^{\mathrm{u}})$, 2 進符号化 τ_x^{b} を定めている組を $(U^{\mathrm{b}}, V^{\mathrm{b}}, D^{\mathrm{b}}, \boldsymbol{A}^{\mathrm{b}}, \|\cdot\|^{\mathrm{b}})$ とする．この時，以下のように 0-1 CSP へのハイブリッド符号化 τ_x^{h} を標準変換として組 $(U, V, D, \boldsymbol{A}, \|\cdot\|)$ で定義する（“h” は hybrid を表す）．ただし X^{o} は順序符号化する変数の集合を表す．

$$
\begin{aligned}
U &= \begin{cases} U^{\mathrm{o}} & (x \in X^{\mathrm{o}}) \\ U^{\mathrm{b}} & (x \notin X^{\mathrm{o}}) \end{cases} \\
V &= \{x\} \\
D(p) &= \{0, 1\} \quad (p \in U) \\
\boldsymbol{A} &= \begin{cases} \boldsymbol{A}^{\mathrm{o}} & (x \in X^{\mathrm{o}}) \\ \boldsymbol{A}^{\mathrm{b}} & (x \notin X^{\mathrm{o}}) \end{cases} \\
\|E\| &= \begin{cases} \|E\|^{\mathrm{o}} & (x \in X^{\mathrm{o}},\ \mathit{Var}(E) \subset X^{\mathrm{o}}) \\ \|E\|^{\mathrm{u}} & (x \in X^{\mathrm{o}},\ \mathit{Var}(E) \not\subset X^{\mathrm{o}}) \\ \|E\|^{\mathrm{b}} & (x \notin X^{\mathrm{o}}) \end{cases}
\end{aligned}
$$

命題 8. 変換 τ_x^{h} は充足同値である．

Proof. τ_x^{h} が命題 1 の条件を満たすことを示せば良い．詳細は省略する． □

6 ハイブリッド符号化の実装

前節で提案したハイブリッド符号化を実装したシステム Hysca を開発した [19]．Hysca は，定義 1 の CSP だけでなく，alldifferent 等のグローバル制約を含んだ一般的な CSP を取り扱える．

Hysca の動作は以下の通りである．

(1) 与えられた CSP を定義 1 の CSP に変換する．
(2) 4.1 節の方法で単純 CSP に変換する．
(3) 5 節の方法で 0-1 CSP に変換する．得られた 0-1 CSP は単純である．
(4) 4.6 節の方法で PB に変換する．
(5) 得られた PB を命題論理ソルバーで解き，解を求める．

(1) には Sugar のプリプロセッサを用いた．Sugar のプリプロセッサは，グローバル制約などを線形式に変換する．(2)–(4) には，Scarab をベースとしたプログラムを開発し，それを用いた．

(5) には，SAT ソルバーである Sat4j をベースとし，PB 制約を直接的に取り扱えるように拡張したソルバーを用いた．したがって 5 節の X^{o} を全変数にした場合，順序符号化を用いて CSP を SAT に符号化している Sugar と同様の動作になる．

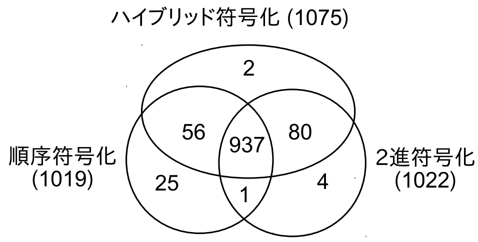

図 1　Hysca での各符号化の求解数

7 実験結果

2009 年の国際 CSP ソルバー競技会 [13] のベンチマーク問題から，外延的制約を含まずかつ連続したドメインの問題 1453 問を使用し，提案するハイブリッド符号化を既存の順序符号化および 2 進符号化と比較し，評価を行った．

使用する CSP ソルバーは Hysca で，実行した計算機の CPU は 3.2GHz, メモリは 16GB，制限時間は 600 秒とした．

表 1 に，各符号化で制限時間内に解けた問題数を問題のシリーズ別に示す．表中，O.E. は順序符号化，B.E. は 2 進符号化，H.E. はハイブリッド符号化を表し，太字の数字は他の方法よりも多い問題数を解いた場合を示している．結果として，提案するハイブリッド符号化が全体として 1075 問を解き，最も優れていた．

図 1 に 3 種類の符号化別に解けた問題数を示す．中央の 937 は，すべての符号化で解けた問題数であり，その左の 56 は, 順序符号化とハイブリッド符号化のみで解けた問題数を示している．この図を見ると，順序符号化で解けて 2 進符号化で解けなかった問題数は $81 = 56 + 21$ だったが，ハイブリッド符号化はそのうちの約 70% に当たる 56 問を解くことに成功していることがわかる．同様に，2 進符号化で解けて順序符号化で解けなかった問題数は $84 = 80 + 4$ だったが，ハイブリッド符号化はそのうちの約 95% に当たる 80 問を解くことに成功している．すなわちハイブリッド符号化は，順序符号化と 2 進符号化のどちらかでしか解くことができなかった 165 問の約 82% に当たる 136 問を解いている．

したがってハイブリッド符号化は，順序符号化と 2

表 1 Hysca での各符号化の求解数 (シリーズ毎)

Series	O.E.	B.E.	**H.E.**
2D Strip Packing (20)	6	5	6
All Interval Series (15)	**15**	8	13
BIBD (83)	**57**	38	38
BMC (15)	15	15	15
Chessboard Coloration (15)	12	12	12
Cumulative Job-Shop (10)	**4**	3	3
Domino (10)	10	10	10
Fischer (25)	3	10	10
Golomb Ruler (28)	17	17	17
Graph Coloring (141)	88	84	88
Haystacks (15)	3	2	3
Job-Shop (76)	58	50	58
Knights (10)	5	**9**	8
Langford (43)	25	21	25
Latin Square (10)	9	7	9
Magic Square (18)	7	7	7
Multi Knapsack (6)	4	6	6
NengFa (7)	5	3	5
Open-Shop (75)	71	49	71
Perfect Square Packing (74)	43	43	**44**
Pigeons (29)	23	23	23
Primes (76)	39	**67**	66
Pseudo-Boolean (363)	269	304	**305**
Queens (25)	16	17	17
RCPSP (78)	78	78	78
Rader Surveillance (65)	65	65	65
Ramsey (16)	5	5	5
Schurr's Lemma (10)	8	**9**	8
Social Golfers (10)	5	**6**	5
Super-solutions (85)	54	49	**55**
Solved (1453)	1019	1022	**1075**

表 2 共通に解けた 937 問の平均 CPU 時間

	O.E.	B.E.	H.E
平均 CPU 時間 (秒)	42.0	33.6	**26.8**

進符号化の互いに不得意な面を補完しているといえる．また表 1 を見ると，Perfect Square Packing, Pseudo-Boolean, Super-solutions のシリーズにおいて，他のどちらの符号化よりも多い問題数を解いている．さらに，すべてで共通に解けた 937 問の平均 CPU 時間を調べると表 2 の通りになり，この点でも最も優れていることが確認できた．

8 おわりに

本論文では，制約充足問題の命題論理への符号化に関し，既存の順序符号化と 2 進符号化を融合させたハイブリッド符号化を提案した．提案したハイブリッド符号化を実装したシステム Hysca を開発し，2009 年国際 CSP ソルバー競技会のベンチマーク問題 1453 問について，既存の符号化と求解数を比較した所，最も優れた性能を示した．

しかし，いくつかの問題シリーズについては，まだ改善の余地が残っており，より良いハイブリッド手法を検討する必要がある．

参考文献

[1] Ignasi Abío, Robert Nieuwenhuis, Albert Oliveras, Enric Rodríguez-Carbonell, and Valentin Mayer-Eichberger. A new look at BDDs for pseudo-Boolean constraints. *Journal of Artificial Intelligence Research*, Vol. 45, pp. 443–480, 2012.

[2] Olivier Bailleux, Yacine Boufkhad, and Olivier Roussel. A translation of pseudo Boolean constraints to SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 2, No. 1-4, pp. 191–200, 2006.

[3] Daniel Le Berre and Anne Parrain. The Sat4j library, release 2.2. *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 7, No. 2-3, pp. 59–64, 2010.

[4] Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, Vol. 185 of *Frontiers in Artificial Intelligence and Applications (FAIA)*. IOS Press, 2009.

[5] James M. Crawford and Andrew B. Baker. Experimental results on the application of satisfiability algorithms to scheduling problems. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI 1994)*, pp. 1092–1097, 1994.

[6] Johan de Kleer. A comparison of ATMS and CSP techniques. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI 1989)*, pp. 290–296, 1989.

[7] Niklas Eén and Niklas Sörensson. Translating pseudo-Boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 2, No. 1-4, pp. 1–26, 2006.

[8] Martin Gebser, Benjamin Kaufmann, André Neumann, and Torsten Schaub. Conflict-driven answer set solving. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pp. 386–392, 2007.

[9] Allen Van Gelder. Another look at graph coloring

via propositional satisfiability. *Discrete Applied Mathematics*, Vol. 156, No. 2, pp. 230–243, 2008.

[10] I. P. Gent and P. Nightingale. A new encoding of alldifferent into SAT. In *Proceedings of the 3rd International Workshop on Modelling and Reformulating Constraint Satisfaction Problems*, 2004.

[11] 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, pp. 57–67, 2010.

[12] Kazuo Iwama and Shuichi Miyazaki. SAT-variable complexity of hard combinatorial problems. In *Proceedings of the IFIP 13th World Computer Congress*, pp. 253–258, 1994.

[13] Christophe Lecoutre, Olivier Roussel, and Marc R. C. van Dongen. Promoting robust black-box solvers through competitions. *Constraints*, Vol. 15, No. 3, pp. 317–326, 2010.

[14] Vasco M. Manquinho, Ruben Martins, and Inês Lynce. Improving unsatisfiability-based algorithms for Boolean optimization. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT 2010), LNCS 6175*, pp. 181–193, 2010.

[15] Vasco M. Manquinho and João P. Marques Silva. On using cutting planes in pseudo-Boolean optimization. *Journal on Satisfiability, Boolean Modeling and Computation*, Vol. 2, No. 1-4, pp. 209–219, 2006.

[16] Amit Metodi and Michael Codish. Compiling finite domain constraints to SAT with BEE. *Theory and Practice of Logic Programming*, Vol. 12, No. 4-5, pp. 465–483, 2012.

[17] 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, pp. 68–76, 2010.

[18] Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

[19] 佐古田淳史. 制約充足問題のハイブリッド符号化に関する研究. Master's thesis, 神戸大学大学院システム情報学研究科, 2015.

[20] Takehide Soh, Naoyuki Tamura, and Mutsunori Banbara. Scarab: A rapid prototyping tool for SAT-based constraint programming systems. In *Proceedings of the 16th International Conference on Theory and Applications of Satisfiability Testing (SAT 2013), LNCS 7962*, pp. 429–436, July 2013.

[21] Mirko Stojadinovic and Filip Maric. mesat: multiple encodings of CSP to SAT. *Constraints*, Vol. 19, No. 4, pp. 380–403, 2014.

[22] Naoyuki Tamura, Mutsunori Banbara, and Takehide Soh. Pbsugar: Compiling pseudo-boolean constraints to SAT with order encoding. In *Proceedings of the 25th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2013)*, pp. 1020–1027, November 2013.

[23] Naoyuki Tamura, Akiko Taga, Satoshi Kitagawa, and Mutsunori Banbara. Compiling finite linear CSP into SAT. In *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP 2006), LNCS 4204*, pp. 590–603, 2006.

[24] Naoyuki Tamura, Akiko Taga, Satoshi Kitagawa, and Mutsunori Banbara. Compiling finite linear CSP into SAT. *Constraints*, Vol. 14, No. 2, pp. 254–272, 2009.

[25] 田村直之, 丹生智也, 番原睦則. SAT 変換に基づく制約ソルバーとその性能評価. コンピュータソフトウェア, Vol. 27, No. 4, pp. 183–196, 2010.

[26] Tomoya Tanjo, Naoyuki Tamura, and Mutsunori Banbara. A compact and efficient SAT-encoding of finite domain CSP. In *Proceedings of the 14th International Conference on Theory and Applications of Satisfiability Testing (SAT 2011), LNCS 6695*, pp. 375–376. Springer, June 2011.

[27] 丹生智也, 田村直之, 番原睦則. 位取り記数法に基づく整数有限領域上の制約充足問題のコンパクトかつ効率的な SAT 符号化. コンピュータソフトウェア, Vol. 30, No. 1, pp. 211–230, 2013.

[28] Toby Walsh. SAT v CSP. In *Proceedings of the 6th International Conference on Principles and Practice of Constraint Programming (CP 2000)*, pp. 441–456, 2000.