特　集

# SATによるシステム検証

## SAT-Based System Verification

番原 睦則
Mutsunori Banbara
神戸大学学術情報基盤センター
Information Science and Technology Center, Kobe University
banbara@kobe-u.ac.jp, http://kaminari.istc.kobe-u.ac.jp/banbara-jp.html

田村 直之
Naoyuki Tamura
（同　　上）
tamura@kobe-u.ac.jp, http://bach.istc.kobe-u.ac.jp/tamura-jp.html

**keywords:** SAT, bounded model checking, combinatorial testing, covering arrays

## 1. はじめに

システム検証とは，ハードウェア／ソフトウェアの信頼性を高めるための手法に関する研究分野である．信頼性の高いシステムを実現するには，設計段階，開発段階において可能な限り欠陥をなくすことが重要である．システムの大規模化・複雑化に伴い，システム検証に関する研究は近年非常に活発になっている．主な検証手法としては，モデル検査，定理証明，組み合わせテスト，高信頼プログラミング言語を用いたアプローチ等がある．モデル検査については，その創始者 Edmund M. Clarke, E. Allen Emerson, Joseph Sifakis が 2007 年度 ACM チューリング賞を受賞している．

命題論理の充足可能性判定 (SAT) は，与えられた命題論理式の充足可能性を判定する問題である．SAT は Cook により最初に NP 完全性が証明された問題である．SAT 問題を解く SAT ソルバーの性能が飛躍的に向上したことをうけて，SAT を多分野に応用する研究が急速に拡大している [井上 10, 藤田 07]．主な応用分野としては，論理合成，プランニング問題，スケジューリング問題，制約充足問題，制約最適化問題，定理証明等がある．

本稿では，SAT をシステム検証へ応用する試みとして，有界モデル検査と組み合わせテストのテストケース自動生成について解説を行う．

有界モデル検査は，SAT を用いた形式的検証手法の一つである．この手法は，従来からの二分決定木による記号モデル検査とは相補的な関係にあり，大規模なシステムに対しても適用できる点が特長である．そのため，回路検証等の分野で実用的に用いられるようになっている．現状で，一万ゲート規模の回路に対し 100 時刻以上の検査が可能とされている．有界モデル検査は，これまで回路検証等のハードウェア検証に関する研究が主流であったが，ここ数年，ソフトウェアへの適用を目指す研究が活発化している．本稿では，有界モデル検査の基本的な検証手法について概観する．

組み合わせテストはソフトウェア／ハードウェアのテスト手法の一つである．この手法の特長は，欠陥の多くは少数のパラメータの組み合わせによって発生するという観測を元に，テストケースの増大を回避し，現実的かつ効果的なテストケースを生成できる点である．組み合わせテスト手法としては，直交表に基づく手法，オールペア法等が広く知られている．近年これらの手法を拡張する試みとして，被覆配列に基づく新しい組み合わせテストの研究が進み，相次いで効率のよいテストケース生成方法が提案されている．本稿では，被覆配列に基づく組み合わせテストの概要を述べた後，SAT を用いたテストケース自動生成について解説する．

以下，2 章で有界モデル検査，3 章で被覆配列に基づく組み合わせテストと SAT を用いたテストケース自動生成について述べ，4 章で本稿をまとめる．

## 2. 有界モデル検査

**モデル検査** (model checking) は，動的なシステムについて，安全性 (safety) や活性 (liveness) などの性質 (property) が成り立つかどうかを検査することが目的である [Clarke 99]．通常，検査すべき性質は LTL (Linear Temporal Logic) [Pnueli 77] や CTL (Computation Tree Logic) [Clarke 82] などの時相論理の論理式で与えられる．

Biere らが提案した**有界モデル検査** (BMC, Bounded Model Checking) は，近年の SAT ソルバーの急速な性能向上を背景として，SAT 技術を用いたモデル検査手法の一つである [Biere 99]．この方法は，従来からの二分決定木 (BDD, Binary Decision Diagram) による方法などとは異った特徴を持ち，より大規模なシステムに対しても適用可能であったため，回路検証等の分野で実用的に用いられるようになった．

BMC では，動的なシステムについて，制限された長さの (すなわち有界の) 実行トレースを命題論理式として記号的に表現し，その命題論理式を SAT ソルバーで検証することにより，性質の検査を行う．

SAT ソルバーが判定するのは与えられた命題論理式の

性質がPのとき ¬Pを SATで解く



充足可能性であるため，BMC で性質の検査を行う場合，性質の否定命題を使用する．

すなわち，与えられた性質の否定命題について SAT ソルバーが充足可能と判定した場合，制限された長さの実行トレース中に，その性質に関する反例が存在することを表す．逆に充足不能と判定した場合，制限された長さの実行トレース中には反例が存在しないことを意味する．この場合，実行トレースの長さを増加させた新たな否定命題を構成し，再び SAT ソルバーによる検証を繰り返す．

BMC は，この意味で性質の反例を探すだけで性質の証明は行わない不完全な手続きであるが，検査すべき実行トレースの長さに上限が存在する等の場合には，完全な手続きとなる．例えば，システムの状態遷移による有向グラフを考え，その直径 (任意の 2 状態間の最短経路のうちの最大長) を上限とすれば良い．

最近の研究では，帰納法や Craig の補間定理を用いることで，性質の証明を行う試みも存在する．これらを含めたより詳細な解説については，よくまとまったサーベイである文献 [Prasad 05, Biere 09] を参照していただきたい．

以下では，具体例を交えながら BMC の手法について説明を進める．

まずシステムの状態が $x_1, x_2, \ldots, x_n$ の $n$ ビットで表されるとし，それらのベクトルを $s = (x_1, x_2, \ldots, x_n)$, $s' = (x'_1, x'_2, \ldots, x'_n)$ 等で表記する．また，$k$ ステップ目 ($k \geq 0$) における各ビットの値を $x_1^k, x_2^k, \ldots, x_n^k$ で表すことにし，それらのベクトルを $s_k = (x_1^k, x_2^k, \ldots, x_n^k)$ 等と表記する．

次に，$I(s)$ により初期状態の条件を表す論理式を与える．以下の例では $x_1 = 0, x_2 = 0$ が初期状態である．

$$I(s) = \neg x_1 \wedge \neg x_2$$

また，ステップ動作による状態遷移を表す論理式を $T(s, s')$ で与える．ここでは，以下の例を考える[*1]．

$$T(s, s') = (x'_1 \leftrightarrow x_2) \wedge (x'_2 \leftrightarrow (\neg x_1 \wedge \neg x_2))$$

この式は，1 ステップ動作後の $x_1$ と $x_2$ の値 ($x'_1$ と $x'_2$) が，動作前の $x_2$ と $\neg x_1 \wedge \neg x_2$ の値にそれぞれ等しいことを表している．これを真理値表の形でまとめると以下のようになる．

| $x_1$ | $x_2$ | $x'_1$ | $x'_2$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 |

これを状態遷移図で表すと図 1 のようになり，0, 1, 2 の三通りの値を周期的に繰り返す 2 ビット・カウンタの例となっている．以下ではカウンタの値を $c$ で表すことにする．

*1 $A \leftrightarrow B$ は同値，すなわち $(A \rightarrow B) \wedge (B \rightarrow A)$ を意味する．

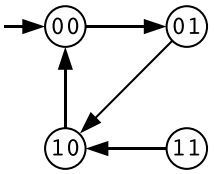


**図 1** 状態遷移図

次に，検査すべき安全性を表す論理式を $P(s)$ で与える．LTL の論理式で表現すると $\mathbf{G}P$ に対応する．ここでは $c \neq 3$ を意味する次の式を用いる．

$$P(s) = \neg(x_1 \wedge x_2)$$

システムが初期状態から $k$ ステップ動作した後の状態 $s_k$ は，$I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1})$ を満たす．また安全性は，すべての $i = 0, 1, \ldots, k$ で $P(s_i)$ が成り立つことを意味している．そこで，反例を探索するため，その否定を付け加えた以下の論理式 $\varphi_k$ を構成する．

$$\varphi_k = I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg P(s_i)$$

いずれかの $k$ に対し $\varphi_k$ が充足可能になれば，安全性の反例が得られる．

したがって，SAT ソルバーを用いて BMC を行う手続きは以下のようになる．まず，$k = 0$ として $\varphi_k$ を構成する．$\varphi_k$ を連言標準形 (CNF) に変換し，SAT ソルバーの入力とする．SAT ソルバーの出力結果が充足可能であればそれが反例となり，充足不能であれば $k$ の値を増加させ，手続きを繰り返す．ただし，$k$ がシステムの直径を越えた所で繰り返しを停止できる [Biere 99].

上記の例の場合，$\varphi_0, \varphi_1, \varphi_2$ のいずれも充足不能であり，常に $c \neq 3$ が成立することがわかる．

一方 $c \neq 2$ を意味する $\neg(x_1 \wedge \neg x_2)$ を $P(s)$ として用いた場合，$\varphi_0$ と $\varphi_1$ は充足不能だが，$\varphi_2$ は充足可能となり $c \neq 2$ は成立しない．

なお，上記の手続き中 $\varphi_k$ を連言標準形に変換する処理が問題となるが，これについては Tseitin 変換の方法を用いれば元の論理式に比例したサイズの充足同値な論理式に変換できることが知られている．Tseitin 変換の詳細については，本特集号中の [田村 10] 等を参照されたい．

### 2·1 活性の検査

前節では安全性の検査を例として取り上げたが，本節ではもう一つの重要な性質である活性の有界モデル検査について説明する．ここでは，LTL で $\mathbf{GF}P$ と表される論理式を対象とする．

活性の検査では，すべての実行トレースにおいて，いつかは必ず与えられた状態に到達することを確認する必要がある．すなわち，与えられた状態に決して到達しない実行トレースが存在すれば，それが反例となる．しかし，そのような実行トレースは無限の長さを持つ．

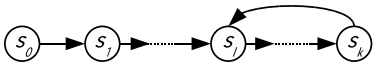


**図 2** $(k,l)$-lasso

そこで，図 2 に示すような $(k,l)$-lasso と呼ばれる状態遷移を考える [Biere 99, Biere 09]．$(k,l)$-lasso では状態 $s_k$ の次の状態が $s_l$ となり，その実行トレースは無限ループになっている．

$(k,l)$-lasso に対する $\varphi_k$ の構成方法は以下のようになる．$l$ は 0 から $k$ のいずれかであるので，条件 $\bigvee_{l=0}^{k} T(s_k,s_l)$ を追加する．また，活性の反例は，すべての状態 $s$ で $\neg P(s)$ となることでなので $\bigwedge_{i=0}^{k} \neg P(s_i)$ を追加する．

$$\varphi_k = I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i,s_{i+1}) \wedge \bigvee_{l=0}^{k} T(s_k,s_l) \wedge \bigwedge_{i=0}^{k} \neg P(s_i)$$

ここで，いずれかの $k$ で $\varphi_k$ が充足可能になれば，反例が得られることになる．

### 2·2 帰納法による性質の証明

通常の BMC では，与えられた性質に対する反例を探索するだけであり，その証明は得られない．理論的には，システムの直径を越えるまで手続きを繰り返せば良いのだが，実用的な問題では直径が大きすぎ実際的には適用できない．

そこで，性質の証明を得る方法として，検査すべき実行トレースの上限をより小さくする工夫，Craig の補間定理を用いる方法，帰納法を用いる方法などが提案されている [Prasad 05, Biere 09].

ここでは，それらのうち帰納法 (induction) を用いる方法について説明する．

通常，帰納法で性質 $P(s)$ を証明する場合，以下の (B) と (I) を示せば良い．

$$\text{(B)}\ \ \forall s.(I(s) \rightarrow P(s))$$
$$\text{(I)}\ \ \forall s.\forall s'.((P(s) \wedge T(s,s')) \rightarrow P(s'))$$

(B) はすべての初期状態 $s$ で性質 $P(s)$ が成り立つことを意味し，(I) は状態 $s$ で $P(s)$ が成立すれば $s$ の次の状態 $s'$ でも必ず性質 $P(s')$ が成立することを意味している．

ここで，通常の BMC と同様に (B) と (I) の否定命題を考え，それらを SAT に変換した命題論理式の充足不能性を示せば良いのだが，上記のままでは条件が弱すぎ，性質の証明を得られない場合が多い．例えば，初期状態から到達不可能な状態 $s_1$, $s_2$ について，$T(s_1,s_2)$, $P(s_1)$, $\neg P(s_2)$ であるとする．この時，(I) の否定命題は充足可能であり性質の証明は得られない．

そこで，Sheeran らは直前の状態だけでなく，それ以前の状態の系列を用いる $k$-帰納法 ($k$-induction) を用いた [Sheeran 00]．$k$-帰納法における (B) と (I) は以下のようになる．

$$\text{(B)}\ \ \forall s_0 \ldots \forall s_k.((I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i,s_{i+1})) \rightarrow \bigwedge_{i=0}^{k} P(s_i))$$

$$\text{(I)}\ \ \forall s_0 \ldots \forall s_{k+1}.((\bigwedge_{i=0}^{k} (P(s_i) \wedge T(s_i,s_{i+1})) \rightarrow P(s_{k+1}))$$

(B) は，初期状態から到達可能な最初の $k+1$ の状態で性質が成立することを意味し，(I) は，連続する長さ $k+1$ の状態で性質が成立するならば次の状態でも必ず性質が成立することを意味している．$k=0$ の場合には通常の帰納法と同一である．

(B) と (I) の否定命題は，それぞれ以下のようになる (存在限量子は省略).

$$\neg\text{(B)}\ \ I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i,s_{i+1}) \wedge \bigvee_{i=0}^{k} \neg P(s_i)$$

$$\neg\text{(I)}\ \ \bigwedge_{i=0}^{k} (P(s_i) \wedge T(s_i,s_{i+1})) \wedge \neg P(s_{k+1})$$

いずれかの $k$ に対して，¬(B) が充足可能であれば性質に対する反例が得られる．この部分は通常の BMC と同様である (¬(B) と $\varphi_k$ が同一である点に注意)．¬(B) が充足不能の場合，¬(I) も充足不能なら性質に対する証明が得られたことになる．

$k$-帰納法は通常の帰納法より強力だが，$k$-帰納法でも性質に対する証明が得られない場合が存在する．例えば，初期状態から到達不可能な状態 $s_1$, $s_2$ について，$T(s_1,s_1)$, $T(s_1,s_2)$, $P(s_1)$, $\neg P(s_2)$ であるとする．この時，任意の $k$ について ¬(I) は充足可能となる．したがって，性質が成立していたとしても，$k$-帰納法による証明は得られない．

そのため文献 [Sheeran 00] では，状態が重複しないという条件を追加している．しかし，変換後の SAT 問題のサイズが大きくなりすぎる問題点がある．MiniSat 作者でもある Eén と Sörensson は，この問題に関して SAT ソルバーのインクリメンタル探索機能を用いる方法を提案している [Eén 03b].

## 3. 被覆配列に基づく組み合わせテストと SAT を用いたテストケース自動生成

ソフトウェア／ハードウェアのテストは，製品を開発する過程において重要な役割を果たしている．しかし，たとえ小規模な製品であっても，総当たりテストはテストケースの増大を招き現実的に実行不可能である．また，ソフトウェア／ハードウェアの大規模化・複雑化に伴い，

必要となるテストケースが増える一方，そのテスト期間については短期化が求められている．

**組み合わせテスト** (combinatorial testing) は，ソフトウェア／ハードウェアのテスト手法の一つである．この手法の特長は，欠陥の多くは少数のパラメータの組み合わせによって発生するという観測を元に，テストケースの増大を回避し，現実的に実行可能であり，かつ効果的なテストケースを生成できる点である．

たとえば，$k$ 個の 2 値入力をもつ回路をテストする場合，$2^k$ 通りのすべてのテストケースを試すにはコストがかかる．そこで，その代わりに，任意の $t$ 個の入力 $(t<k)$ に対して，$2^t$ 通りあるそれらの値の組み合わせすべてを含むテストケースを試したいとする．この場合に知りたいのは，必要となるテストケース数の最小値である．この問題は，被覆配列 $CA(b;t,k,2)$ が存在する最小の $b$ を求める問題に帰着できる．

オールペア法 (あるいは，ペアワイズ法) は，$t=2$ の被覆配列に基づく組み合わせテスト手法である．商用を含め現存する組み合わせテスト・ツールの多くは，このオールペア法を用いている．

### 3·1 被 覆 配 列

**被覆配列** (covering array) [*2]$CA(b;t,k,g)$ とは，$b\times k$ 配列 ($b$ 行 $k$ 列) であり，各要素は $\{0,1,2,\ldots,g-1\}$ のいずれかの値を取る．そして，どの $t$ 個の列 $(1\le t\le k)$ についても，全部で $g^t$ 通りあるそれらの値の組み合わせすべてが少なくとも一つ出現する．ここでは，$t$ を強さ，$k$ を因子，$g$ を水準と呼ぶことにする．目的は $CA(b;t,k,g)$ が存在する最小の $b$ を見つけることである．

**【例 1】** 組み合わせテストにおける一つのテストケースが，$CA(b;t,k,g)$ の各行に対応しており，テストケース数が $b$ の値に対応する．例として，強さ $t=3$，因子 $k=5$，水準 $g=2$ に対するテストケースの生成，すなわち $CA(b;3,5,2)$ を構成する問題を考える．図 3 に，この問題の最適解 ($b=10$) を示す．最初の 3 列について，異なる 0 と 1 の組み合わせをボールド体で記している．全部で $2^3$ 通りあるそれらの値の組み合わせすべてが，少なくとも一つ出現していることがわかる．他のどの 3 列の組み合わせについても同様の性質を満たす．

以下の定義は，論文 [Hnich 06] に基づいている．

**【定義 1】** 被覆配列 $CA(b;t,k,g)$ とは，以下の性質を満たす $b\times k$ 配列 $A=(a_{ij})$ である．

- $a_{ij}\in\mathbb{Z}_g=\{0,1,2,\ldots,g-1\}$
- 任意の異なる $t$ 個の列 $1\le c_1\le c_2\le\ldots\le c_t\le k$，および任意の値の組 $(x_1,x_2,\ldots,x_t)\in\mathbb{Z}_g^t$ に対して，$x_i=a_{rc_i}(\forall i;1\le i\le t)$ を満たす行 $r$ が少なくとも一つ存在する．

**【定義 2】** 被覆配列数 $CAN(t,k,g)$ とは，$CA(b;t,k,g)$ が存在する最小の $b$ である．

$$CAN(t,k,g)=\min\{b\mid CA(b;t,k,g)\}$$

| | | | | |
|---|---|---|---|---|
| **0** | **0** | **0** | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| **0** | **0** | **1** | 0 | 0 |
| **0** | **1** | **0** | 0 | 0 |
| **0** | **1** | **1** | 1 | 1 |
| **1** | **0** | **0** | 0 | 0 |
| **1** | **0** | **1** | 1 | 1 |
| **1** | **1** | **0** | 1 | 1 |
| **1** | **1** | **1** | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

**図 3** $CA(10;3,5,2)$．どの 3 個の列についても，全部で $2^3$ 通りあるそれらの値の組み合わせすべてが出現している．

被覆配列数 $(2\le t\le 6)$ の最新情報は，Charles Colbourn によって Web 上に公開されている [Colbourn 09]．表 1 と表 2 に，$2\le t\le 3$ および $k,g$ の小さな値に対して，これまでに知られている $CAN(t,k,g)$ の最新の値 [Chateauneuf 02, Colbourn 09] を示す．各表とも，$(k,g)$ 成分が整数 $m$ の場合は $CAN(t,k,g)=m$ を表し，$m,n$ の場合は $m\le CAN(t,k,g)\le n$ を表している．

被覆配列の構成方法については，組み合わせテストへの応用を視野に入れ，$CA(b;t,k,g)$ に対してより小さな $b$ を効率よく求める方法を中心に研究が進んでいる．主なアプローチは以下の通りである．

- 群論等を用いた数学的手法 [Williams 00, Chateauneuf 02, Hartman 04, Meagher 05]
- 貪欲法 [Cohen 97, Lei 98, Tung 00]
- 局所探索法 [Cohen 03, Nurmela 04]
- 制約充足問題としての定式化・SAT 符号化 [Hnich 05, Hnich 06]

数学的手法と貪欲法は，問題の規模に関係なく良質な被覆配列を効率よく構成することができる．一方，これまでに知られている被覆配列数の上限の多くは，局所探索法によって求められている．しかし，貪欲法と比較して，良質な上限を求めるには非常に時間がかかるという短所がある．Hnich らによって提案された制約充足問題としての定式化・SAT 符号化は，比較的新しい研究結果である．現在のところ，その有効性は小・中規模な問題に対してのみ示されている．しかし，実用的な組み合わせテストを実現するために必要な “水準の異なる因子”，“禁則処理” 等 [Hartman 04] への対応が比較的容易であるという長所がある．

被覆配列数および被覆配列の構成方法の詳細については，サーベイ論文 [Sloane 93, Colbourn 04] を参照していただきたい．

以下，被覆配列の構成方法について，制約充足問題としての定式化を示した後，SAT 符号化を用いた構成方法について解説する．

*2 本稿では covering array の訳語として “被覆配列” を用いる．

**表 1** 既知の $CAN(2,k,g)$

| $k\backslash g$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 3 | 4 | 9 | 16 | 25 | 36 | 49 | 64 | 81 |
| 4 | 5 | 9 | 16 | 25 | 37 | 49 | 64 | 81 |
| 5 | 6 | 11 | 16 | 25 | 37,39 | 49 | 64 | 81 |
| 6 | 6 | 12 | 19 | 25 | 37,41 | 49 | 64 | 81 |
| 7 | 6 | 12 | 19,21 | 29 | 37,42 | 49 | 64 | 81 |
| 8 | 6 | 12,13 | 19,22 | 29,33 | 39,42 | 49 | 64 | 81 |
| 9 | 6 | 12,13 | 19,23 | 29,35 | 39,46 | 52,59 | 64 | 81 |
| 10 | 6 | 12,14 | 19,24 | 29,36 | 39,51 | 52,61 | 67,72 | 81 |
| 11 | 7 | 12,15 | 19,24 | 29,38 | 39,54 | 52,67 | 67,78 | 84,105 |
| 12 | 7 | 12,15 | 19,24 | 29,38 | 39,56 | 52,73 | 67,85 | 84,105 |
| 13 | 7 | 12,15 | 19,25 | 29,40 | 39,57 | 52,76 | 67,92 | 84,105 |
| 14 | 7 | 12,15 | 19,27 | 29,41 | 39,59 | 52,79 | 67,99 | 84,113 |

**表 2** 既知の $CAN(3,k,g)$

| $k\backslash g$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 4 | 8 | 27 | 64 | 125 | 216 | 343 | 512 |
| 5 | 10 | 28,33 | 64 | 125 | 222,240 | 343 | 512 |
| 6 | 12 | 33 | 64 | 125 | 222,258 | 343 | 512 |
| 7 | 12 | 36,40 | 76,88 | 125,180 | 222,293 | 343 | 512 |
| 8 | 12 | 36,42 | 76,88 | 145,185 | 222,304 | 343 | 512 |
| 9 | 12 | 36,45 | 76,112 | 145,185 | 234,379 | 343,472 | 512 |
| 10 | 12 | 36,45 | 76,112 | 145,185 | 234,393 | 364,479 | 512 |
| 11 | 12 | 36,45 | 76,121 | 145,225 | 234,463 | 364,637 | 536,960 |
| 12 | 14,15 | 36,45 | 76,121 | 145,225 | 234,463 | 364,637 | 536,960 |
| 13 | 14,16 | 36,51 | 76,124 | 145,245 | 234,503 | 364,637 | 536,960 |
| 14 | 14,16 | 36,51 | 76,124 | 145,245 | 234,503 | 364,637 | 536,960 |
| 15 | 14,17 | 36,57 | 76,124 | 145,245 | 234,514 | 364,637 | 536,960 |
| 16 | 14,17 | 36,60 | 76,124 | 145,245 | 234,514 | 364,637 | 536,960 |

### 3·2 制約充足問題としての定式化

被覆配列の構成問題は，制約充足問題 (CSP, Constraint Satisfaction Problem) として定式化することができる．ここでは，Hnich らの定式化 [Hnich 06] を説明する．

まず，以下の二つの行列を考える．

- 基本行列
  整数変数を要素とする $b\times k$ 行列．各整数変数 $x_{r,i}$ $(1\le r\le b, 1\le i\le k)$ は，被覆配列の各要素の値を表し，そのドメインは $x_{r,i}\in\{0,1,2,\ldots,g-1\}$ である．すなわち $x_{r,i}=m$ は $CA(b;t,k,g)$ の $(r,i)$ 成分が $m$ であることを表す．
- 拡張行列
  整数変数を要素とする $b\times\binom{k}{t}$ 行列．各列は $t$ 個の因子の可能な組み合わせの一つを表す．例 1 の場合，因子の組み合わせは $\binom{k}{t}=\binom{5}{3}=10$ 通りあるため，各列は以下の $T$ 中の 1 つの組を表す．
  $$T=\{(1,2,3),(1,2,4),(1,2,5),(1,3,4),(1,3,5),\\(1,4,5),(2,3,4),(2,3,5),(2,4,5),(3,4,5)\}$$
  各整数変数 $y_{r,i'}$ $(1\le r\le b, 1\le i'\le\binom{k}{t})$ は，基本行列における $t$ 個の変数の組を表し，そのドメインは $y_{r,i'}\in\{0,1,2,\ldots,g^t-1\}$ である．例 1 の場合，整数変数 $y_{r,(i,j,\ell)}$ $(1\le r\le b,\ 1\le i<j<\ell\le k)$ は，基本行列の変数の組 $(x_{r,i},x_{r,j},x_{r,\ell})$ を表し，そのドメインは $y_{r,(i,j,\ell)}\in\{0,\ldots,7\}$ となる．すなわち $y_{r,(i,j,\ell)}=7$ は $x_{r,i}=1, x_{r,j}=1, x_{r,\ell}=1$ を表す．図 4 に図 3 を拡張行列で表現したものを示す．

あとは，以下の二つを制約条件として記述することで，被覆配列 $CA(b;t,k,g)$ の構成問題を制約充足問題として定式化できる．

- カバレッジ制約
  どの $t$ 個の列 $(1\le t\le k)$ についても，全部で $g^t$ 通りあるそれらの値の組み合わせすべてが少なくとも一つ出現する．
- チャネリング制約
  拡張行列の各変数は，基本行列の対応する $t$ 個の変数と整合する．

カバレッジ制約は，大域制約 (global constraint) の一つである大域基数制約 (global cardinality constraint) [Régin 96] を用いて簡単に表現できる．すなわち，この大域基数制約を拡張行列の各列に適用することにより，各列中に 0〜$g^t-1$ の値が少なくとも一つ出現するという制約条件 (カバレッジ制約の下限) を表現する[*3]．拡張行列の各列は $t$ 個の因子の可能な組み合わせの一つを表してい

*3 大域基数制約を用いれば，各列中に 0〜$g^t-1$ の値が高々 $b-g^t+1$ 個出現するという制約条件 (カバレッジ制約の上限) も表現できる．ただし，この条件は省略可能である．

| (1,2,3) | (1,2,4) | (1,2,5) | (1,3,4) | (1,3,5) | (1,4,5) | (2,3,4) | (2,3,5) | (2,4,5) | (3,4,5) |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 2 | 1 | 0 | 2 | 2 |
| **1** | 0 | 0 | 2 | 2 | 0 | 2 | 2 | 0 | 4 |
| **2** | 2 | 2 | 0 | 0 | 0 | 4 | 4 | 4 | 0 |
| **3** | 3 | 3 | 3 | 3 | 3 | 7 | 7 | 7 | 7 |
| **4** | 4 | 4 | 4 | 4 | 4 | 0 | 0 | 0 | 0 |
| **5** | 5 | 5 | 7 | 7 | 7 | 3 | 3 | 3 | 7 |
| **6** | 7 | 7 | 5 | 5 | 7 | 5 | 5 | 7 | 3 |
| **7** | 6 | 7 | 6 | 7 | 5 | 6 | 7 | 5 | 5 |
| 7 | 7 | 6 | 7 | 6 | 6 | 7 | 6 | 6 | 6 |

**図 4** 図 3 の $CA(10;3,5,2)$ を拡張行列で表現．各列は横線の上に示した因子の組を表している．

るため，全部で $g^t$ 通りある値の組み合わせすべてを含むことを保証できる．チャネリング制約については，内包的表現と外延的表現の二通りが可能である．例 1 の場合，その内包的表現は $y_{r,(i,j,\ell)} = 4x_{r,i} + 2x_{r,j} + x_{r,\ell}$ となる．また，外延的表現を用いる場合は以下のようになる．

$$(y_{r,(i,j,\ell)}, x_{r,i}, x_{r,j}, x_{r,\ell}) \in \{(0,0,0,0),(1,0,0,1),\\ (2,0,1,0),(3,0,1,1),(4,1,0,0),\\ (5,1,0,1),(6,1,1,0),(7,1,1,1)\}$$

### 3·3 SAT 符号化を用いた被覆配列の構成

SAT 符号化とは，元の問題を SAT 問題に変換して，SAT ソルバーを用いて解を求める方法である．ここでは，3·2 節で示した被覆配列の CSP 表現を SAT 問題に変換する方法について説明する [Hnich 06].

まず，基本行列の各整数変数 $x_{r,i}$ と各定数 $v \in \mathbb{Z}_g$ に対して，ブール変数 $p(x_{r,i} = v)$ を導入する．$p(x_{r,i} = v)$ は $x_{r,i} = v$ を表す．同様に，拡張行列の各整数変数 $y_{r,i'}$ と各定数 $w \in \mathbb{Z}_{g^t}$ に対して，ブール変数 $p(y_{r,i'} = w)$ を導入する．$p(y_{r,i'} = w)$ は $y_{r,i'} = w$ を表す．

あとは，すべての $1 \le r \le b$, $1 \le i \le k$, $0 \le v < v' \le g-1$, $1 \le i' \le \binom{k}{t}$, $0 \le w < w' \le g^t - 1$ に対して，以下の節を生成すればよい．

- 整数変数の変換

$$\bigvee_v p(x_{r,i} = v) \tag{1}$$

$$\neg p(x_{r,i} = v) \vee \neg p(x_{r,i} = v') \tag{2}$$

$$\bigvee_w p(y_{r,i'} = w) \tag{3}$$

$$\neg p(y_{r,i'} = w) \vee \neg p(y_{r,i'} = w') \tag{4}$$

- カバレッジ制約の変換

$$\bigvee_r p(y_{r,i'} = w) \tag{5}$$

- チャネリング制約の変換

$$\neg p(y_{r,i'} = w) \vee p(x_{r,i} = v) \tag{6}$$

(ただし，この節は $y_{r,i'} = w$ と $x_{r,i} = v$ が整合する $r$,$i$,$i'$,$v$,$w$ に対してのみ生成する)

(1) と (3) は，それぞれ $x_{r,i}$ と $y_{r,i'}$ が少なくとも一つの値をとることを表す at-least-one 節である．(2) と (4) は，それぞれ $x_{r,i}$ と $y_{r,i'}$ が同時に二つ以上の値をとらないことを表す at-most-one 節である．(5) はカバレッジ制約，すなわち，拡張行列の各列中に 0〜$g^t - 1$ の値が少なくとも一つ現れることを表す節である．(6) について，例 1 の場合，そのチャネリング制約は以下のように変換される (ただし，$w = 4v_1 + 2v_2 + v_3$).

$$\neg p(y_{r,(i,j,\ell)} = w) \vee p(x_{r,i} = v_1)$$
$$\neg p(y_{r,(i,j,\ell)} = w) \vee p(x_{r,j} = v_2)$$
$$\neg p(y_{r,(i,j,\ell)} = w) \vee p(x_{r,\ell} = v_3)$$

3·2 節の CSP 表現と異なる点は，カバレッジ制約の上限に関する制約がないことである．この制約は SAT で表現することは可能だが困難であるという点から省略されている．

なお，本節で使用した SAT 符号化法は支持符号化法 (support encoding) と呼ばれる方法に基づいている．SAT 符号化の詳細については，今回の特集号に田村らによる解説 [田村 10] があるので，参照していただきたい．

### 3·4 被覆配列数：SAT を用いた結果

Hnich らは，3·3 節の SAT 符号化を使って，被覆配列の構成問題を SAT 問題に変換し，確率的 SAT ソルバー (walksat の改良版 [Selman 94]) で求解する実行実験を行っている．その結果，小・中規模の被覆配列に対して，当時知られていた被覆配列数の上限の多くと同じ値を得ることに成功した．彼らが見つけた上限 $CAN(3,7,3) \le 40$ は現在でも最良値である．

筆者らは，SAT を用いたテストケース自動生成に関する今後の可能性を探るために，Hnich らと異なる実行実験を行った．まず，3·2 節の CSP 表現に対して，対称解を除去するための辞書順序制約を追加した．その後，順序符号化法 (order encoding) [Tamura 06, Tamura 09] と呼ばれる SAT 符号化法を使って SAT 問題に変換し，高速な系統的 SAT ソルバー MiniSat [Eén 03a] で求解した．ベンチマーク問題としては，強さ $t = 3$ の被覆配列のうち被覆配列数が未決定である小規模な問題 6 問を使用し

**表 3** $CA(b;3,k,g)$ の実行結果

| $k$ | $g$ | $b$ | SAT/UNSAT | 変数の数 | 節の数 | CPU 時間 (秒) |
|---|---|---|---|---|---|---|
| 12 | 2 | 14 | UNSAT | 207400 | 800696 | 45158 |
| 12 | 2 | **15** | SAT | 235786 | 922129 | 27 |
| 13 | 2 | 14 | UNSAT | 268580 | 1038296 | 29221 |
| 13 | 2 | 15 | UNSAT | 305353 | 1195810 | 103113 |
| 13 | 2 | **16** | SAT | 344178 | 1363512 | 336 |
| 14 | 2 | 14 | UNSAT | 340790 | 1318872 | 10802 |
| 14 | 2 | 15 | UNSAT | 387464 | 1518999 | 21722 |
| 14 | 2 | **16** | SAT | 436740 | 1732058 | 1868 |
| 15 | 2 | 14 | UNSAT | 424947 | 1646001 | 5896 |
| 15 | 2 | 15 | UNSAT | 483162 | 1895817 | 9600 |
| 15 | 2 | 17 | SAT | 609291 | 2443692 | 246179 |
| 16 | 2 | 14 | UNSAT | 521968 | 2023260 | 3267 |
| 16 | 2 | 15 | UNSAT | 593490 | 2330385 | 4336 |
| 5 | 3 | 31 | UNSAT | 141370 | 668938 | 63 |
| 5 | 3 | 32 | UNSAT | 152891 | 724098 | 145 |
| 5 | 3 | **33** | SAT | 164155 | 777547 | 34322 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

**図 5** CA(16;3,14,2)

た：$12 \leq k \leq 16$ に対する $CA(b;3,k,2)$ と $CA(b;3,5,3)$. 表 3 に実験結果を示す．左から因子 $k$，水準 $g$，サイズ $b$，SAT と UNSAT の別，生成された SAT 問題の変数の数，節の数，MiniSat の CPU 時間を示している．実験環境は Mac Pro (Quad), Intel Xeon 3.2GHz, 16GB メモリである．実験の結果，$CAN(3,12,2)$, $CAN(3,13,2)$, $CAN(3,14,2)$, $CAN(3,5,3)$ について，既知の最良の上限が最適値であることを証明することができた．しかし，$b$ が最適値 (SAT)，最適値-1(UNSAT) に対する問題は難しく，求解に非常に時間がかかる．求解速度の向上には，新たな探索空間の枝狩り手法の導入，生成される SAT 問題のサイズ縮小化等が必要不可欠と考える．最後に，$CA(16;3,14,2)$ の構成例を図 5 に示す．

## 4. ま と め

本稿では，SAT をシステム検証に応用する試みとして，有界モデル検査，被覆配列に基づく組み合わせテストおよび SAT によるテストケース自動生成に関する研究について解説を行った．

有界モデル検査は SAT を用いた形式的検証手法の一つである．有界モデル検査は，性質の反例を探すことはできるが，性質の証明は行えないという意味で不完全な手続きであるといえる．最近の研究では，帰納法や Craig の補間定理を用いることで，性質の証明を行う方法が提案されている．一方，有界モデル検査については，これまで回路検証等のハードウェア検証に関する研究が主流であったが，ここ数年，ソフトウェアへの適用を目指す研究が国内外で活発化している．本稿では，具体例を交えながら，有界モデル検査の基本的な検証手法について述べた．また，性質の証明を得る方法として，帰納法を用いる方法を紹介した．

組み合わせテストはソフトウェア／ハードウェアのテスト手法の一つである．組み合わせテスト手法としては，オールペア法 (強さ $t = 2$ の被覆配列に基づく組み合わせテスト手法) が広く普及しているが，近年，より大きな $t > 2$ への拡張を目指す研究が活発化している．また，実用的な組み合わせテストを実現するために必要な “水準の異なる因子”，“禁則処理” 等に関する研究も進みつつある．SAT を用いたテストケース自動生成は，SAT ソルバーの性能の進歩により，最近注目されつつある SAT の応用である．本稿では，被覆配列に基づく組み合わせテストに関する研究を概観し，SAT を用いたテストケース自動生成について，その具体的な手法および簡単な実験結果を紹介した．スペースの関係上言及できなかったが，被覆配列の構成を制約充足問題として定式化する場合，辞書順序制約等を用いた対称性の除去 (symmetry breaking)

が非常に有効であることが知られている [Hnich 06].

SAT をシステム検証に応用する試みは，本稿で紹介した以外にも数多く存在する．例えば，回路のテスト手法として知られるテストパターン自動生成 (ATPG: Automatic Test Pattern Generation) [Drechsler 09] は，SAT の重要な応用の一つである．

## ◇ 参 考 文 献 ◇


[Biere 99] Biere, A., Cimatti, A., Clarke, E. M., and Zhu, Y.: Symbolic Model Checking without BDDs, in *Proceedings of the 5th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'99), LNCS 1579*, pp. 193–207 (1999)

[Biere 09] Biere, A.: *Handbook of Satisfiability*, chapter 14: Bounded Model Checking, pp. 457–481, IOS Press (2009)

[Chateauneuf 02] Chateauneuf, M. A. and Kreher, D. L.: On the State of Strength-Three Covering Arrays, *Journal of Combinatorial Designs*, Vol. 10, No. 4, pp. 217–238 (2002)

[Clarke 82] Clarke, E. M. and Emerson, E. A.: Design and Synthesis of Symchronization Skeletons Using Branching-Time Temporal Logic, in *Proceedings of the Workshop on Logic Programs, LNCS 131* (1982)

[Clarke 99] Clarke, E. M., Grumberg, O., and Peled, D.: *Model Checking*, MIT Press (1999)

[Cohen 97] Cohen, D. M., Dalal, S. R., Fredman, M. L., and Patton, G. C.: The AETG System: An Approach to Testing Based on Combinatiorial Design, *IEEE Trans. Software Eng.*, Vol. 23, No. 7, pp. 437–444 (1997)

[Cohen 03] Cohen, M. B., Gibbons, P. B., Mugridge, W. B., and Colbourn, C. J.: Constructing Test Suites for Interaction Testing, in *Proceedings of the 25th International Conference on Software Engineering*, pp. 38–48 (2003)

[Colbourn 04] Colbourn, C. J.: Combinatorial Aspects of Covering Arrays, *Le Matematiche (Catania)*, Vol. 58, pp. 121–167 (2004)

[Colbourn 09] Colbourn, C. J.: Covering Array Tables for t=2,3,4,5,6, http://www.public.asu.edu/~ccolbou/src/tabby/catable.html (2009), Last Accessed on Oct 11 2009

[Drechsler 09] Drechsler, R., Eggersglüß, S., Fey, G., and Tille, D.: *Test Pattern Generation using Boolean Proof Engines*, Springer (2009)

[Eén 03a] 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)*, pp. 502–518 (2003)

[Eén 03b] Eén, N. and Sörensson, N.: Temporal Induction by Incremental SAT Solving, in *Proceedings of the 1st International Workshop on Bounded Model Checking (BMC), ENTCS 89* (2003)

[Hartman 04] Hartman, A. and Raskin, L.: Problems and Algorithms for Covering Arrays, *Discrete Mathematics*, Vol. 284, No. 1-3, pp. 149–156 (2004)

[Hnich 05] Hnich, B., Prestwich, S. D., and Selensky, E.: Constraint-Based Approaches to the Covering Test Problem, in Faltings, B., Petcu, A., Fages, F., and Rossi, F. eds., *Recent Advances in Constraints, Joint ERCIM/CoLogNet International Workshop on Constraint Solving and Constraint Logic Programming, CSCLP 2004*, Vol. 3419 of *Lecture Notes in Computer Science*, Springer (2005)

[Hnich 06] Hnich, B., Prestwich, S. D., Selensky, E., and Smith, B. M.: Constraint Models for the Covering Test Problem, *Constraints*, Vol. 11, No. 2-3, pp. 199–219 (2006)

[井上 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)

[Lei 98] Lei, Y. and Tai, K.-C.: In-Parameter-Order: A Test Generation Strategy for Pairwise Testing, in *Proceedings of 3rd IEEE International Symposium on High-Assurance Systems Engineering (HASE'98)*, pp. 254–261 (1998)

[Meagher 05] Meagher, K. and Stevens, B.: Group Construction of Covering Arrays, *Journal of Combinatorial Designs*, Vol. 13, No. 1, pp. 70–77 (2005)

[Nurmela 04] Nurmela, K. J.: Upper Bounds for Covering Arrays by Tabu Search, *Discrete Applied Mathematics*, Vol. 138, No. 1-2, pp. 143–152 (2004)

[Pnueli 77] Pnueli, A.: The Temporal Logic of Programs, in *Proceedings of IEEE Symposium on Foundations of Computer Science* (1977)

[Prasad 05] Prasad, M., Biere, A., and Gupta, A.: A Survey of Recent Advances in SAT-based Formal Verification, *Software Tools for Technology Transfer*, Vol. 7, No. 2, pp. 156–173 (2005)

[Régin 96] Régin, J.-C.: Generalized Arc Consistency for Global Cardinality Constraint, in *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI'96)*, Vol. 1, pp. 209–215, AAAI Press / The MIT Press (1996)

[Selman 94] Selman, B., Kautz, H. A., and Cohen, B.: Noise Strategies for Improving Local Search, in *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI'94)*, pp. 337–343 (1994)

[Sheeran 00] Sheeran, M., Shingh, S., and Støalmarck, G.: Checking Safety Properties using Induction and a SAT-solver, in *Proceedings of the 3rd International Conference on Formal Methods in Computer-Aided Design (FMCAD), LNCS 1954*, pp. 108–125 (2000)

[Sloane 93] Sloane, N. J. A.: Covering Arrays and Intersecting Codes, *Journal of Combinatorial Designs*, Vol. 1, pp. 51–63 (1993)

[Tamura 06] Tamura, N., Taga, A., Kitagawa, S., and Banbara, M.: Compiling Finite Linear CSP into SAT, in *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP 2006)*, pp. 590–603 (2006)

[Tamura 09] Tamura, N., Taga, A., Kitagawa, S., and Banbara, M.: Compiling finite linear CSP into SAT, *Constraints*, Vol. 14, No. 2, pp. 254–272 (2009)

[田村 10] 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)

[Tung 00] Tung, Y.-W. and Aldiwan, W.: Automating Test Case Generation for the New Generation Missionsoftware System, in *Proceedings of IEEE Aerospace Conference*, pp. 431–437 (2000)

[Williams 00] Williams, A. W.: Determination of Test Configurations for Pair-Wise Interaction Coverage, in *Proceedings of 13th International Conference on Testing Communicating Systems (TestCom 2000)*, pp. 59–74 (2000)

[藤田 07] 藤田 昌宏：SAT アルゴリズムの最新動向, 電子情報通信学会誌, Vol. 90, No. 12, pp. 1067–1072 (2007)


〔担当委員：××○○〕

19YY 年 MM 月 DD 日 受理

## 著 者 紹 介


**番原 睦則**

1994 年神戸大学理学部数学科卒業．1996 年同大学大学院自然科学研究科博士課程前期数学専攻修了．1996 年国立奈良工業高等専門学校助手．1998 年同校講師．2003 年より神戸大学学術情報基盤センター学術情報処理研究部門講師．2007 年同校准教授．博士 (工学)．論理プログラミング，線形論理，制約プログラミング，SAT などに興味をもつ．日本ソフトウェア科学会，情報処理学会会員．

**田村 直之**(正会員)

1980 年神戸大学理学部物理学科卒業．1985 年同大学院自然科学研究科博士課程システム科学専攻修了．学術博士．1985 年日本 IBM 東京基礎研究所入社．1988 年神戸大学工学部勤務．2003 年より神戸大学学術情報基盤センター学術情報処理研究部門教授．制約プログラミング，SAT，線形論理，論理プログラミングなどに興味をもつ．日本ソフトウェア科学会，情報処理学会会員．