論　文

SMT を用いた制約付きのロケーティングアレイの生成について

金　浩[†,††a)]　　崔　銀惠[††]　　土屋　達弘[†]

On the generation of constrained locating arrays using an SMT solver

Hao JIN[†,††a)], Eunhye CHOI[††], and Tatsuhiro TSUCHIYA[†]

あらまし　ソフトウェアに対するインタラクションテストにおいて，ロケーティングアレイをテスト集合として用いることで，故障検出だけでなくその特定も可能となる．本研究では，テストパラメータ上の制約を考慮したロケーティングアレイの生成を，SMT ソルバを用いて行う方法について説明する．

キーワード　ロケーティングアレイ，組合せテスト，SMT ソルバ

1. まえがき

ソフトウェアの開発において，その不具合，つまりフォールトを検出するためのデバッグに莫大なコストが費やされている．ソフトウェアシステムが日々複雑化している現在において，高効率な不具合検出手法はその重要性を著しく表し，具体的な検出手法の一つとして，組合せテスト（combinatorial testing）は注目を集めている．組合せテストはあらゆる k 個のパラメータに対し，それらが取りうる値の組合せ（interaction）をすべて網羅することによって，フォールトに繋がるパラメータ値の組合せを検出できる．その根拠として，Kuhn らが 2002 年 [1] 及び 2004 年 [2] に発表された論文により，大多数のソフトウェアの不具合はパラメータ数 k が 6 以下の値の組み合わせにより生じることが挙げられる．組合せテストを利用することで，複数のテストケース（test case）が生成され，それらの集合をテストスイート（test suite）と名付ける．それぞれのテストケースは各パラメータに値を振り当てることであり，一つのベクトルとなる．そして，それらのベクトルの集合がテストスイートであり，マトリックスの形式で表す．

組合せテストの核となる部分は「k 個のパラメータの値の組み合わせを全て網羅する」であることは明らかであり，このような網羅に対する評価を k-way カバレッジ（k-way coverage）と呼ぶ．しかし，全てのパラメータ値の組合せを網羅したとしても，テスト集合のサイズが大き過ぎては，そのアドバンテージは現れない．いかに k-way カバレッジの条件を満たしつつ，テストケース数を削減し，テストスイートのサイズを圧縮するかが焦点である．当問題は組合せ最適化問題の類に属し，先行研究では，様々な組合せテスト生成手法が提案されている．[3] を始めとする貪欲法による生成方法もあり，他にも [4] のような SAT ソルバを用いた生成法も存在する．

さらなる効果を求めるため，組合せテストによる検出機能のみならず，フォールトを特定する機能をも持つロケーティングアレイ（locating array）が提案されている [5]．組合せテストの特徴はパラメータ値の組合せを網羅することであり，その結果として生成されたテストスイートの中，たとえフォルス（false）になるテストケースを発見しても，そのテストケースが含む全ての組合せがフォールティインタラクション（faulty　interaction）の候補であるため，そのさらなる特定は難しい．そこで，テストケースを増加することで，同一テストケースに含まれる組合せ間でも，他のテストケースと比較することにより，特定できるようなテストスイート，ロケーティングアレイが望ましい．フォールティインタラクション数が 1 であり，

† 大阪大学情報科学研究科，吹田市
Graduate School of Information Science and Technology, Osaka University, Yamadaoka 1–5, Suita-shi, Osaka, 565–0871 Japan
†† 産業総合技術研究所，池田市
Information Technology Research Institute, AIST Kansai, Midorigaoka 1–8–31, Ikeda-Shi, Osaka, 563–8577 Japan
a) E-mail: k-kou@ist.osaka-u.ac.jp

2-way カバレッジを満たすロケーティングアレイを (1, 2)-locating array と呼ぶ.

本論文の章構成は以下の通りである．**2.** で制約付きロケーティングアレイについて述べ，**3.** では SMT 問題への変換について説明する．**4.** では提案手法に基づく実験を行い，実行時の動作結果について考察する．最後に **5.** では，研究の結論と今後の展望について述べる．

2. 制約付きロケーティングアレイ

1. で述べたように，ロケーティングアレイは組合せテストのカバー特性を保ちつつ，テストケースを増やすことにより，さらなる特定機能を得たテストスイートである．これまでに，ロケーティングアレイについての研究は [6] と [7] などが存在する．しかし，対象システム（SUT，System Under Testing）の固有制約を考慮したロケーティングアレイの研究はまだ少ない．一般的に，ほとんどのシステムはその仕様により，入力パラメータまたはコンポーネント間で固有の制限，制約（constraint）が存在する．これらを考慮することで，ロケーティングアレイはより実用化となる．本章ではそのような制約を考慮したロケーティングアレイ，(1, 2)-constrained locating array(CLA) の生成について述べる．

2.1 基本概念

- SUT モデル

制約付きロケーティングアレイに適するシステム SUT を以下のように定義する．SUT はタプル $\langle \mathcal{F}, \mathcal{S}, \phi \rangle$ で表す，そのうち，$\mathcal{F} = \{F_1, F_2, ..., F_k\}$ と定義し，SUT における k 個の入力パラメータを表す；各パラメータにおける変域の集合を $\mathcal{S} = \{S_1, S_2, ..., S_k\}$ と定義する；$\phi : S_1 \times ... \times S_k \rightarrow \{true, false\}$ は SUT モデルの固有制約を表す．各パラメータ変域には，二値以上の値を持ち，それらを 0 からの番号を振り当てることで，記述上の便利を図る，すなわち，$S_i = \{0, 1, ..., |S_i - 1|\}$ のように表す．以上のように定義することから，各テストケース $\boldsymbol{\sigma}$ は $S_1 \times S_2 \times ... \times S_k$ の形をした一つのベクトルとなる．N 個のテストケースを含むテストスイート A は，$N \times k$ のアレイである．

- 無効な値の組合せ

システム定義の ϕ により，特定なパラメータの値の組合せは有効ではないことがある．このような組合せは無効な値の組合せ（invalid interaction）と呼び，テスト設計では，テストケースに出現してはいけない．例を挙げると，システム $\langle \mathcal{F}, \mathcal{S}, \phi \rangle$ において，$|\mathcal{F}| = 5, |S_1| = ... = |S_5| = 2$，パラメータ F_1 及びパラメータ F_2 は以下のような制約 ϕ_1 が存在する：

$$\phi_1 : F_1 = 0 \rightarrow F_2 = 0$$

すなわち，(F_1, F_2) の組合せ $\{(0,0), (0,1), (1,0), (1,1)\}$ のなか，$(1,0)$ は ϕ_1 と矛盾しているため，テストケースの中では出現できない．故に，このシステムにおいては，$(1,0)$ は無効な値の組合せの一つとなる．$\{(0,0), (0,1), (1,1)\}$ のようなインタラクションは有効な値の組合せ（valid interaction，以降 VI とする）であり，すべての有効な二値の組合せ集合を $\mathcal{VI}_2$ と示す．そして，このようにパラメータに値を振り当てることを $\sigma(F_i, s_i)$ と示す．テストケースにおいては，VI のみを含むものが有効である．テストスイート A のなか，有効なインタラクション T を含むテストケースの集合を $\rho_A(T)$ と表す．

- 区別不可能なインタラクションペア

テストケースを実行し，その結果からフォルティインタラクションを特定するには，以下の式を満たす必要がある：

$$\forall T_1, T_2 \in \mathcal{VI}_2; T_1 \neq T_2 \Leftrightarrow \rho_A(T_1) \neq \rho_A(T_2) \quad (1)$$

故に，任意の異なる 2 つのインタラクションは，少なくとも一つのテストケースでは，同時に出現してはいけない．しかし，異なる VI 間でも，固有制約により，上記の論理式を満たせない場合がある：

$$T_1, T_2 \in \mathcal{VI}_2 \wedge T_1 \neq T_2 \wedge \rho_A(T_1) = \rho_A(T_2)$$

このような 2 つの VI は (1, 2)-CLA において特定不可能であるため，区別不能なインタラクションペア（indistinguishable pair，以降 IP とする）と定義する．前例同様に，$|\mathcal{F}| = 5, |S_1| = ... = |S_5| = 2$ のようなシステムを考える．ここで，インタラクション $(\sigma(F_1, 0), \sigma(F_2, 0))$ と $(\sigma(F_1, 0), \sigma(F_3, 0))$ は VI とする．システムの制約 ϕ に以下のような 2 つの制約を想定する：

$$\phi_2 : F_2 = 0 \rightarrow F_3 = 0 \quad (2)$$

$$\phi_3 : F_3 = 0 \rightarrow F_2 = 0 \quad (3)$$

つまり，パラメータ F_2, F_3 においては，片方が 0 の値を取る場合，もう片方も必ず 0 の値を取るという

表 1　SUT : 携帯電話

$\mathcal{F}$	F_1: Display	F_2: Email Viewer	F_3: Camera	F_4: Video Camera	F_5: Video Ringtones
$\mathcal{S}$	0:16MC 1:8MC 2:BW	0:Graphical 1:Text 2:None	0:2MP 1:1MP 2:None	0:Yes 1:No	0:Yes 1:No
ϕ	$\phi_1 : F_2 = 0 \rightarrow F_3 \neq 2$ $\phi_2 : F_3 = 0 \rightarrow F_1 \neq 2$ $\phi_3 : F_2 = 0 \rightarrow F_3 \neq 0$ $\phi_4 : F_1 = 1 \rightarrow F_3 \neq 0$ $\phi_5 : F_4 = 0 \rightarrow (F_3 \neq 2 \wedge F_1 \neq 2)$ $\phi_6 : F_5 = 0 \rightarrow F_4 = 0$ $\phi_7 : \neg(F_1 = 0 \wedge F_2 = 1 \wedge F_3 = 0)$				

表 2　無効なインタラクション

$((\sigma(F_1, 2), (\sigma(F_2, 0))$
$((\sigma(F_1, 1), (\sigma(F_3, 0))$
$((\sigma(F_1, 2), (\sigma(F_3, 0))$
$((\sigma(F_1, 2), (\sigma(F_4, 0))$
$((\sigma(F_1, 2), (\sigma(F_5, 0))$
$((\sigma(F_2, 0), (\sigma(F_3, 0))$
$((\sigma(F_2, 1), (\sigma(F_3, 0))$
$((\sigma(F_3, 2), (\sigma(F_4, 0))$
$((\sigma(F_3, 2), (\sigma(F_5, 0))$
$((\sigma(F_4, 1), (\sigma(F_5, 0))$

表 3　区別不可能なインタラクションペア

$((\sigma(F_1, 0), (\sigma(F_3, 0))$	$\sim$	$((\sigma(F_2, 2), (\sigma(F_3, 0))$
$((\sigma(F_1, 2), (\sigma(F_4, 1))$	$\sim$	$((\sigma(F_1, 2), (\sigma(F_5, 1))$
$((\sigma(F_3, 2), (\sigma(F_4, 1))$	$\sim$	$((\sigma(F_3, 2), (\sigma(F_5, 1))$

ことである．ϕ_2, ϕ_3 により，$(\sigma(F_1, 0), \sigma(F_2, 0))$ と $(\sigma(F_1, 0), \sigma(F_3, 0))$ の 2 つの VI は必ず同一のテストケースに出現しなければならない．故に，この両者は IP である．すべての VI に対し，2 つのインタラクションを選択し，それらが IP ではない場合は，少なくとも一つのテストケースに同時出現させないことで，(1) を満たすことができる．

2.2　適 応 事 例

実用例として，[8] に現れた携帯電話の例を利用する．SUT モデルは表 1 に示されている，$|\mathcal{F}| = 5, |\mathcal{S}| = 3^3 2^2, |\phi| = 7$．これらをもとに，**2.1** で記述した順にインタラクションの有効性及びインタラクション間の区別可能性を確かめていく．

システムパラメータの値集合 $\mathcal{S}$ に対し，すべての 2-way インタラクションの集合を $\mathcal{I}_2$ とする．そして，制約 ϕ から得られるすべての VI を $\mathcal{VI}_2$ と示す．システム制約 ϕ による無効なインタラクションを表 **2** にて示す．$\mathcal{VI}_2$ に対して，IP は表 **3** のように得られる，これらを取り除き，残りのインタラクションペアを少なくとも一つのテストケースにおいて，同時出現しないように設計することで，(1, 2)-CLA を得ることができる．表 **4** にてその実例を示す．

表 4　SUT: (1, 2)-CLA

CLA	F_1	F_2	F_3	F_4	F_5
σ_1	0	0	1	0	0
σ_2	0	0	2	1	1
σ_3	0	1	1	0	1
σ_4	0	2	0	0	0
σ_5	0	2	0	0	1
σ_6	0	2	0	1	1
σ_7	0	2	2	1	1
σ_8	1	0	1	0	1
σ_9	1	0	1	1	1
σ_{10}	1	1	1	0	0
σ_{11}	1	1	2	1	1
σ_{12}	1	2	1	0	0
σ_{13}	2	1	1	1	1
σ_{14}	2	1	2	1	1
σ_{15}	2	2	1	1	1

3.　SMT 問題への変換

SMT(satisfiability modulo theories) は SAT 問題 (satisfiability problems) に基づき，論理変数だけでなく，整数及びそれらの計算記号などをも扱える理論である．SMT ソルバを利用することで，羅列した命題の充足可能な変数の値（解）を求められる．SAT/SMT ソルバに関する研究はすでに捗っていて，高速に解を求めることができる．故に，SMT ソルバのエンコーディングの容易さ，実行の素早さを利用することで，(1, 2)-CLA を求めることが効果的だと考えられる．本章では，**2.2** にて討論された SUT モデルを対象にし，SMT-LIB の標準に基づき論議する．

3.1　エンコーディング

SMT ソルバを用いて問題を解くために，まず変数を宣言しなければならない：

```
(declare-const f1 Int)
```

上記のように SUT の 5 つのパラメータを定義する．パラメータの値に関しては，断言（assertion）を加えることにより, その取りうる値を制限できる：

```
(assert (and (>= f1 0) (<= f1 2))
```

上記の断言から観察できるが，断言はすべてポーランド記法により記述される．変数宣言を実行した後，各パラメータの値の範囲，そして制約を順次に書き出す．

入力されたすべての断言に対し，真となる解が存在するのならば，SMT ソルバは「SAT」の結果を出し，

各変数の値を提示する．しかし，真となる解が存在しないのならば，つまり，断言間にて矛盾が存在するのならば，SMT ソルバは「UNSAT」の結果を提示する．

3.1.1 無効なインタラクション

定義に基づき，無効なインタラクションは SUT モデルの固有制約 ϕ と矛盾しているため出現する．現に SMT ソルバに変数の宣言，値の範囲，固有制約を入力したとする．すべての 2-way インタラクション $\mathcal{I}_2$ において，入力済みの断言と合わせ「UNSAT」の結果が出現した場合，そのインタラクションは無効だと判断できる：

(assert (and (= f1 2) (= f2 0)))

すべてのインタラクションに対し，上記の断言をそれぞれ実行した結果，表 **2** の 10 個の無効なインタラクションを特定できる．これら無効なインタラクションの集合を $\mathcal{I}_2$ から取り除き，$\mathcal{VI}_2$ を求められる．

3.1.2 区別不能なインタラクションペア

$\mathcal{VI}_2$ を得た後，集合中の任意な異なる 2 つのインタラクション T_1, T_2 に対し，その区別可能性を確かめる．無効なインタラクションと同様，先に SMT ソルバに変数宣言，値範囲及び固有制約を入力する．定義によると，区別可能性を確かめる方法は，同じテストケースに同時に出現させないことで，それが可能であれば，両者は区別可能である．この特性から，以下のような命題を書き出せる：

$$((F_1 = s_0) \wedge (F_2 = s_0)) \oplus ((F_1 = s_0) \wedge (F_3 = s_0))$$

上記の命題を SMT-LIB 標準に基づき，書き換える：

```
(assert (xor   (and (= f1 0) (= f2 0))
               (and (= f1 0) (= f3 0))))
```

この式をすべての $\mathcal{VI}_2$ ペアに応用する．「UNSAT」の結果を出すペアは IP である．これらを取り除き，残りのペアは区別可能のペアである，$\mathcal{DP}$ と記す．

3.1.3 テストスイート構成

3.2 Symmetry Breaking

4. 実験・評価

5. 結　　論

- 最終原稿の提出に関しては，各ソサイエティの「投稿のしおり」を参照してください．
- ソース・ファイルはできるだけ 1 本のファイルにまとめてください．
- 著者独自のマクロを記述したファイルや文献，図の EPS ファイルなどを忘れていないかご確認願います．

文　　献

[1] D.E. クヌース，改訂新版 TEX ブック，アスキー出版局，東京，1992.
[2] 磯崎秀樹，LATEX 自由自在，サイエンス社，東京，1992.
[3] 藤田眞作，化学者・生化学者のための LATEX—パソコンによる論文作成の手引，東京化学同人，東京，1993.
[4] 阿瀬はる美，てくてく TEX，アスキー出版局，東京，1994.
[5] S. von Bechtolsheim, TEX in Practice, Springer-Verlag, New York, 1993.
[6] N. Walsh, Making TEX Work, O'Reilly & Associates, Sebastopol, 1994.
[7] D. Salomon, The Advanced TEX book, Springer-Verlag, New York, 1995.
[8] phone example

付　　録

1. PDF の作成方法と A4 用紙への出力

- PDF に書き出すには二通りの方法があります．

（1） dvipdfmx を使って PDF に変換する（以下では段幅の関係で折り返します）．

```
dvipdfmx -p 182mm,257mm -x 1in -y 1in
 -o file.pdf file.dvi
```

用紙サイズとして `jisb5` というオプションが使えるかもしれません．オプションの `-x 1in -y 1in` は省略できます．

（2） まず，dvips を使用して，ps に書き出します．

```
dvips -Pprinter -t b5 -O 0in,0in
  -o file.ps file.dvi
```

`printer` には，使用するプリンタ名を記述します．オプションの `-O 0in,0in` は省略できます．

次に Acrobat Distiller で PDF に変換します．

- `dvips` を使用して A4 用紙に出力する場合のパラメータはおおよそ以下のような設定になります．

```
dvips -Pprinter -t a4 -O 14mm,20mm file.dvi
```

`printer` には使用するプリンタ名を記述します．オプションの `-t a4` は省略できます．

- Windows 上の dvout を利用して，用紙の左右天地中央に版面を自動配置する場合は以下のようにします．

（1） dvout を起動します．

（2） メニューバーにある Option の中の Setup

Parameters... を選択します．

（3） 「DVIOUT のプロパティ」というダイアログが開くので，Paper というタブを選択します．

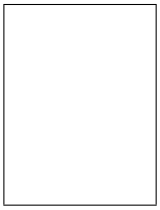
（4） Options という枠の中に Horizontal centering と Vertical centering というチェックボックスがあるので，それぞれチェックした後に Save ボタンを押します．

（5） この設定を行った後に dvi ファイルを開くと，版面が用紙の中心に配置されます．

2. 削除したコマンド

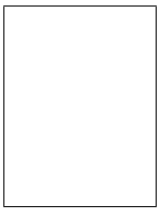
本誌の体裁に必要のないコマンドは削除しています．削除したコマンドは，`\part`，`\theindex`，`\tableofcontents`，`\titlepage`，ページスタイルを変更するオプション（`headings`，`myheadings`）などです．

（平成 xx 年 xx 月 xx 日受付）



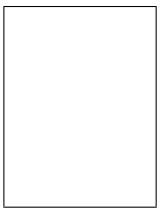
電子　花子　（正員）

1996 東北一大学情報工学科卒．1999 東京第一大学工学部工学部助手．某システムの研究に従事．



情報　太郎　（正員）

1995 大阪一大学工学科卒．1997 同大大学院工学研究科修士課程了．1998 大阪（株）入社．某コンピュータ応用の研究に従事．ABC 学会会員．



通信　次郎

1980 九州一大学理工学部卒．1981 大阪（株）入社．現在 ATT 日本研究所に所属．

Abstract IEICE (The Institute of Electronics, Information and Communication Engineers) provides a pLaTeX 2_ε class file, named `ieicej.cls` (ver. 3.0), for the IEICE Transactions. This document describes how to use the class file, and also makes some remarks about typesetting a manuscript by using the pLaTeX 2_ε. The design is based on ASCII Japanese pLaTeX 2_ε.

Key words pLaTeX 2_ε class file, typesetting, math formulas