様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　３０　年　　６　月　２２　日現在

機関番号：　１４４０１
研究種目：　基盤研究(C)（一般）
研究期間：　2015～2017
課題番号：　１５Ｋ００９８

研究課題名（和文）不具合特定能力を持つ実用的な組み合わせインタラクションテストの実現

研究課題名（英文）Practical combinatorial interaction testing with fault locating capability

研究代表者
土屋　達弘（Tsuchiya, Tatsuhiro）

大阪大学・情報科学研究科・教授

研究者番号：３０２８３７４０

交付決定額（研究期間全体）：（直接経費）　　3,600,000円

研究成果の概要（和文）：本研究では，ソフトウェアシステムに対する組み合わせインタラクションテスト（CIT）を発展させ，高不具合検出能力と低実行コストに加え，不具合の特定能力を持ち合わせるテストの自動設計実現を目的として，関連するテスト生成技術の開発を行った．特に，ロケーティングアレイという数学的構造に注目し，この構造の持つ組み合わせの網羅性と不具合検出特性を現実のテスト問題に利用できるようにした．具体的には，その定義を拡張するとともに，自動的に生成するアルゴリズムを開発した．さらに，いくつかの最小ロケーティングアレイの生成に成功し，開発したヒューリスティックな生成法との比較を行った．

研究成果の概要（英文）：The purpose of this research was to extend Combinatorial Interaction Testing (CIT) to add the capability of fault detection to it. We focused on locating arrays as an enabling technique for this. However, the original definition of locating arrays does not assume the existence of constraints among test parameters, which prohibited locating arrays from being used in real-world CIT. In this research we proposed an extension of locating arrays aiming at addressing this problem. We showed some mathematical properties of this new notion. In addition, we developed a few algorithms that automatically generate locating arrays. Using one of the algorithms, we were successful in finding the minimum (optimal) locating arrays for several cases. These optimal results were used as baseline in the evaluation of the other generation algorithms.

研究分野：　ソフトウェアテスト

キーワード：　ソフトウェアテスト　組み合わせインタラクションテスト

１．研究開始当初の背景

ソフトウェアシステムの開発では，十分なテストの実施と，テスト工程にかかる費用の低減が強く求められる．組み合わせインタラクションテスト（CIT）は，この矛盾した要求を実現する有望な手法として，導入が進んでいる．

テストにより不具合が検出された場合，原因を特定してシステムを修正することが必須である．しかし，現状の CIT では組み合わせの網羅のみを目的としており，不具合が顕在化した場合（テストケースがフェイルした場合）に原因特定に有用なテスト設計については，まだ研究の初期段階にあった．この目的に関して，研究開始時において注目すべき概念としては，ロケーティングアレイがあった[1]．これは，CIT を実現するだけでなく，各テストのパス/フェイルの結果から，不具合の原因となる組み合わせを特定できる性質をもったテスト集合のことである．

ロケーティングアレイに関する既存研究はその存在条件の数学的解析が主で，生成アルゴリズムについての研究はほとんどなかった．また実用を考慮した場合，禁則条件を考慮する必要がある．禁則条件とは，あるパラメータで特定の値を選んだ場合は，別のパラメータにおいて選ぶことのできる値が制限をうけるといった，テストケースが満たすべき条件を指す．従来のロケーティングアレイは禁則を考慮しておらず，それを直接実際のテスト工程に用いることは困難であった．

２．研究の目的

本研究の目的は，不具合特定能力も持つ実用的な CIT 用テスト自動設計の実現である．より具体的には，以下の三つの課題の達成を目的とする．

(1) まず，実用上の制約を反映した問題定式化である．現状のロケーティングアレイの理論では，現実のテストを反映したモデル化がなされていない．禁則条件という実際のテスト工程における制約を適切にモデル化した上でテスト生成問題の定式化を行う．つまり，ロケーティングアレイを再定義し，不具合特定可能な CIT 用テストに形式的かつ実用的な定義を与える．

(2) 次に，新しいロケーティングアレイの定義に従って，不具合特定可能な CIT 用テストを生成するアルゴリズムを開発する．このとき，現実的な時間でできるだけ小さいテスト集合を求めることが最大の課題となる．

(3) さらに，開発したアルゴリズムの性能評価を行う．この際，上記の実用的なテスト生成アルゴリズムとは別に，時間がかかってもよいのでできるだけ小さいテストを生成し，評価のベースラインとする．

３．研究の方法

(1) 実用上の制約を反映した問題定式化に関しては，制約が存在する場合に特定できない不具合が生じることが予想されていた．具体的には，制約のために常に同時にしかテストできない入力の組み合わせのペアについては，テストのパス．フェイルの情報からはペアのいずれに不具合があるかは特定できない．この考え方を形式化し，制約を考慮した上で従来のロケーティングアレイを拡張するような概念を考案する．

(2) テストを生成するアルゴリズムについては，まず，現状のロケーティングアレイの定義に基づき，これを自動的に生成するアルゴリズムを開発する．その上で，その知見に基づき，新しいロケーティングアレイを生成するアルゴリズムを開発する．

(3) できるだけ小さいテスト集合の作成によるアルゴリズムの評価では，ブール式の充足可能性判定問題（SAT）に対するソルバーを用いることで，そのようなテスト集合の作成を実現する．

４．研究成果

(1) 実用上の制約を反映した問題定式化については，制約付きロケーティングアレイ(Constrained Locating Arrays (CLAs))の概念を新たに定義した．具体的な例として，テストパラメータ F1～F5 から構成され，F1～F3 が 0，1，2 の 3 値，F4，F5 が 0，1 の 2 値からなり，以下の禁則条件が存在するシステムを考える．

禁則 1: F2 = 0 ならば F1 ≠ 2
禁則 2: F3 = 0 ならば F1 ≠ 2
禁則 3: F2 = 0 ならば F3 ≠ 0
禁則 4: F1 = 1 ならば F3 ≠ 0
禁則 5: F4 = 0 ならば (F3 ≠ 2 かつ F1 ≠ 2)
禁則 6: F5 = 0 ならば F4 = 0
禁則 7: (F1 = 0 かつ F2 = 1 かつ F3 = 0) ではない

（この例は，ソフトウェアプロダクトラインにおける携帯電話の開発をモデル化したものである．）このとき，F1＝0, F3=0 という組み合わせと，F2=2, F3=0 という組み合わせのいずれかに不具合が存在する場合，どちらが不具合かをテスト結果からは判定することはできない．例えば，(F1,…,F5) = (0,1,0,0,0)や(1,2,0,0,0)などの，両者のうち一方のみを含むようなテストケースは，禁則に違反するため実行することができない．

この観察に基づき，不具合の原因が限定できないような組み合わせ同士を「区別不能」であると定義した．その上で，任意の区別可能な組み合わせのペアについて，不具合の原因がそれらのどちら側であったとしても，原因として特定できるようなテスト集合を，制約付きロケーティングアレイと定義した．定義より，禁則がない場合，制約付きロケーティングアレイはロケーティングアレイに一致する．以下の図に，上記のシステム例において，パラメータ 2 個の値の組み合わせが不具合の原因である場合に，不具合を特定可能

な制約付きロケーティングアレイを示す.

0	0	0	0	0
0	0	1	1	1
0	0	2	0	1
0	1	1	0	0
0	2	0	0	1
1	0	1	0	1
1	0	2	1	1
1	1	0	0	1
1	1	2	1	0
1	2	0	1	0
2	0	2	0	0
2	1	1	1	1
2	2	0	1	0
2	2	1	0	0
2	2	2	1	1

(2)制約付きロケーティングアレイの生成アルゴリズムを，まず，通常のロケーティングアレイの生成アルゴリズムを開発し，それを改変する形で設計した.

まず，通常のロケーティングアレイの生成のため，充足可能性判定を利用する方法を開発した．これは，求めたいテスト集合の大きさ（テストの数）を固定し，テスト集合（アレイ）の各要素の値をブール変数で表現した上で，ロケーティングアレイが満たすべき性質をブール式で表現することで，性質を満たすテスト集合が存在するかどうかの判定を，ブール式の充足可能性判定問題に還元する．これを実現するツールとして，Scalab を用いた．これは制約問題を Scala に基づく独自の言語で簡潔に記述することを実現するものであり，充足可能性判定問題への変換と求解も同時に実行することができる．ブール式を真にする解，すなわち，ブール変数への真理値割り当てが存在すれば，それが求めるロケーティングアレイを表す.

さらに，焼きなまし法を利用して，より高速に，かつ，大きい問題例に適用可能なロケーティングアレイの生成法を開発した．これは，ランダムにアレイを作成した後，繰り返しランダムにテストとパラメータを選択してその値を変化させ，徐々にロケーティングアレイに近づけていく方法である．ロケーティングアレイに近づくよう探索を導けるように，現在の解に対する評価関数を適切に設計した．具体的には，まだテスト集合に現れていない組み合わせの数と，テスト実行の結果からは不具合原因と限定しきれない組み合わせの数を評価するようにした．これらの値がともに 0 になれば，ロケーティングアレイが得られたことになる．実験の結果，上記の充足可能性判定を用いる手法より，はるかに大きな規模の問題に適用可能であることが分かった.

これらの通常のロケーティングアレイの生成手法をふまえて，制約付きロケーティングアレイの生成アルゴリズムの開発を行った．与えられた大きさの制約付きロケーティングアレイが存在するかという問題を充足可能性判定問題として表現することで，一般的なソルバーを用いて制約付きロケーティングアレイを求める方法を開発した．焼きなまし法の利用に関しては，予め区別不能な組み合わせペアを列挙しておき，それらを除くすべての組み合わせに関して不具合の特定が可能になった場合に値が 0 になるように評価関数を設計することで，制約付きロケーティングアレイについても生成が可能なことが分かった.

また，これらの成果とは平行して，2 分決定図の一種である ZDD というデータ構造を用いることで，莫大な数の組み合わせを効率的に操作できることが分かった．これを通常の CIT のテストケース生成に用いることで，これまでできなかった，パラメータ数の多い組み合わせを完全網羅するテスト集合の設計に成功した．この手法のロケーティングアレイの生成への応用について，今後研究を進める.

(3) できるだけ小さいテスト集合の生成によるアルゴリズムの評価では，まず，通常のロケーティングアレイを対象に，(2)で説明した充足可能性判定による生成手法を利用して最小のテスト集合の生成を実現した．具体的には，存在を調べるテスト数を徐々に増加させていき，解が最初に得られた時，その解が最小のロケーティングアレイの存在を示すことになる．これまで特殊な場合以外では，最小のロケーティングアレイについてはほとんど何も分かっておらず，実際にそのような最小のアレイを初めて得ることができた.

制約付きロケーティングアレイについても，充足可能性判定を用いる生成手法を利用し，同様な方法で最小のものを生成することに成功した．具体的には，CIT 分野の過去の文献などで挙げられている１０程度の問題例に対し，最小の制約付きロケーティングアレイを生成できた.

これらの最適な制約付き/制約なしロケーティングアレイに比べて，焼きなまし法によって得られたものは，やや大きい傾向にあることが現在までに分かっている．ただし，問題例が小さい場合は，最適なものが得られる場合も多い．一方で，一つ一つテストケースを追加し，最終的にロケーティングアレイを完成させる貪欲手法[2]と比較すると，焼きなまし法で得られた解はかなり最適値に近い．より詳細な評価は今後の課題である.

＜引用文献＞

[1] Colbourn et al., Locating and detecting arrays for interaction faults, J. Combin. Optimz. 15, 2008.
[2] Nagamoto et al., Locating a faulty interaction in pair-wise testing, PRDC 2014 (Nov. 2014).

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 3 件）

① Tatsuya Konishi, Hideharu Kojima, Hiroyuki Nakagawa, Tatsuhiro Tsuchiya, Finding Minimum Locating Arrays Using a SAT Solver, Proceedings of 10th IEEE International Conference on Software Testing, Verification and Validation Workshops, 査読有, 2017, pp.276-277, DOI: 10.1109/ICSTW.2017.49

② Teru Ohashi, Tatsuhiro Tsuchiya, Generating High Strength Suites for Combinatorial Interaction Testing Using ZDD-Based Graph Algorithms, Proceedings of 22nd IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'17), 査読有, 2017, pp.78-85, DOI: 10.1109/PRDC.2017.19

③ 土屋達弘, ソフトウェアと安全性, 信頼性学会学会誌「信頼性」, 査読無, Vol.38, No.2, 2016, pp.80-85. DOI:10.11348/reajshinrai.38.2_80

〔学会発表〕（計 7 件）

① 土屋達弘, ブール論理に基づく, 情報システムのテスト・検証のためのアプローチ, 人工知能学会人工知能基本問題研究会 (SIG-FPAI), 2018 年 3 月.

② 小西達也, 小島英春, 中川博之, 土屋達弘,焼きなまし法によるロケーティングアレイの生成, 電子情報通信学会ディペンダブルコンピューティング研究会, 2018 年 2 月.

③ 小西達也, 小島英春, 中川博之, 土屋達弘, SAT ソルバを使用したロケーティングアレイの生成手法について, 電子情報通信学会システム数理と応用研究会, 2017 年 11 月.

④ 大橋輝, 土屋達弘, ZDD アルゴリズムを用いた高強度テストケース生成法, 電子情報通信学会ディペンダブルコンピューティング研究会, 2016 年 10 月.

⑤ 大橋輝, 土屋達弘, 貪欲法による組み合わせテスト集合生成における高速化について, 電子情報通信学会ディペンダブルコンピューティング研究会, 2015 年 10 月.

⑥ 大橋輝, 土屋達弘, 組み合わせテスト集合生成におけるテストケース候補数の最適化について, 第 14 回情報科学技術フォーラム, 2015 年 9 月.

⑦ 土屋達弘, ソフトウェアテスト用テストケース生成における２分決定図を用いた制約処理, 電子情報通信学会ディペンダブルコンピューティング研究会, 2015 年 6 月.

〔図書〕（計 0 件）

〔産業財産権〕

○出願状況（計 0 件）

名称：
発明者：
権利者：
種類：
番号：
出願年月日：
国内外の別：

○取得状況（計 0 件）

名称：
発明者：
権利者：
種類：
番号：
取得年月日：
国内外の別：

〔その他〕
ホームページ等

６．研究組織

(1)研究代表者
土屋 達弘（Tsuchiya, Tatsuhiro）
大阪大学・大学院情報科学研究科・教授
研究者番号：30283740

(2)研究分担者
（ ）

研究者番号：

(3)連携研究者
（ ）

研究者番号：

(4)研究協力者
（ ）