Vol.2014-MPS-100 No.15
2014/9/25

# 形式仕様に基づくテストパターン生成への GA の適用

杉原拓[†1]　劉少英[†1]　佐藤裕二[†1]

形式仕様に基づいたソフトウェア開発において，テストパターンの自動生成に関しては既にいくつかの手法が提案されている．しかし，これまでに提案された手法は，仕様が正しく実現できていることを確認するためのものがほとんどであり，プログラムの全てのパスを通過するテストパターン自動生成の問題に関しては未解決である．本稿では，確率的探索手法である遺伝的アルゴリズムを用いたテストパターン生成手法を提案する．テストケースの集合を染色体として定義し，多次元ベクトルを用いた突然変異を提案することで，従来手法に比べてプログラムのパスの網羅率を大きく向上するテストパターン自動生成が実現できる可能性を示す．

## 1. はじめに

形式仕様に基づいたソフトウェア開発において，テストパターンの自動生成に関しては既にいくつかの手法が提案されている．しかし，これまでに提案された手法は，仕様が正しく実現できていることを確認するためのものがほとんどであり，プログラムの全てのパスを通過するテストパターン自動生成の問題に関しては未解決である．例えば，全てのパスを通過するテストパターン自動生成の研究としては確率的探索手法(“Vibration method”)により，従来のテストパターン生成手法 Pairwise 法よりもパス網羅率の精度を向上させることに成功した例が報告されている[1].

また，遺伝的アルゴリズム（GA）を用いたテストパターン生成手法では，特定のパスを通過するテストケース生成するために標準的 GA を用いた例が報告されている[2]．本稿では，全てのパスを通過するテストパターン自動生成手法として，テストケースの集合を染色体として定義し，多次元ベクトルを用いた突然変異を用いた GA の適用を提案することで，従来手法に比べてプログラムのパスの網羅率を大きく向上できる可能性を示す．

## 2. GA の設計

### 2.1 染色体の定義

テストケースを要素とした一次元染色体を定義する．染色体の構成を図 1 に示す．テストケースは全ての入力変数を含み，染色体の定義長はプログラムのパスの総数に一致させる．パスの総数が不明の場合は，冗長性を持たせて少し長めの定義長とする．

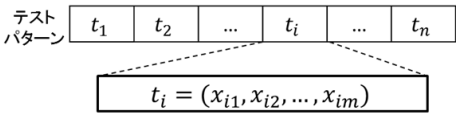


図 1　染色体の構成

### 2.2 評価値

今回の実験は，テストケースによって通ったプログラム中のパスを区別できるという前提でテストパターン生成を行う．最適解は全てのパスを網羅するテストパターンとなる．テストケースを入力することでプログラムのパスを通過させる．テストケースがプログラムのパスを通過したとき，そのテストパターンがそれ以前に通過しているパスの集合$P = \{p_1, p_2, \ldots, p_k\}$の各要素との比較を行う．その結果，新たなパスと判明した場合そのパスを$p_{k+1}$とし，通過したパスの集合に加える．プログラムのパスの総数に対する通過したパスの数の割合をテストパターンの評価値として与え，評価値が 1 となったときに全てのパスを網羅するテストパターンを生成したものとし，処理を終了する．

### 2.3 遺伝的操作

#### 2.3.1 交叉

二点交叉，一様交叉の 2 つの交叉手法について，交叉率，突然変異率をパラメータとして入力変数が 2 つの場合と 3 つの場合について予備実験を行った．その結果，一様交叉を用いた手法において安定した結果が得られたため，以下の評価実験では一様交叉を用いて実験を行う．

#### 2.3.2 突然変異

突然変異は，テストケース$t_i(x_{i1}, x_{i2}, \ldots x_{in})$を$n$次元ベクトルと捉え，変異ベクトル$\vec{m}(x_{m1}, x_{m2}, \ldots x_{mn})$を付加することで，新たなベクトルを生成して突然変異後のテストケース$t'_i(x'_{i1}, x'_{i2}, \ldots x'_{in})$とする．入力変数が 2 つの場合の例を図 2 に示す．次元が増えた場合でも差分ベクトルの大きさと向きの 2 つを決めることで，変異ベクトルを生成できる．

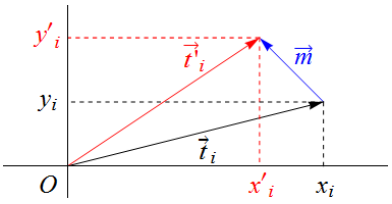


図 2　入力変数が 2 つの場合における突然変異

## 3. 評価実験

### 3.1 評価手法

提案する手法によるテストパターン生成の妥当性を評価するために，パスが全て独立であるプログラム，パスが独立でないプログラムの 2 種類について実験を行う．2 種類のプログラムについて一例を図 3 に示す．独立なパスは分岐が明らかであり，仕様通りの動作のみを行うためパス

†1 法政大学
Hosei University

の総数がわかっている条件での実験である．独立でないパスは仕様外の動作をするため分岐が複雑でありパスの総数がわからない条件での実験である．どちらの実験においても入力変数は整数型としており，入力変数の値によって通過するパスが異なるプログラムである．入力変数にはパスを通過するための前提条件があり，その条件を満たさない入力変数の組み合わせをもつテストケースはどのパスも通過しないものとする．

パスが独立なプログラムにおける実験では提案する突然変異手法と単純突然変異の比較を行う．単純突然変異では，入力変数 1~3 つの値を変数の値の範囲内でランダムに変化させるものである．パスが独立でないプログラムにおける実験ではパスの総数がわからないため，評価値は発見したパスの数を与え，最大評価値がある一定の世代数を経ても更新されない場合に処理を終了する．今回の実験では 500 世代の間これにより，全てのパスを網羅するテストパターンを確実に生成することはできないが準最適解を得ることが可能である．また、両実験共に比較実験を繰り返し行うことで GA の実行パラメータを改良した．最も良い結果を得られたパラメータを表 1 に示す．

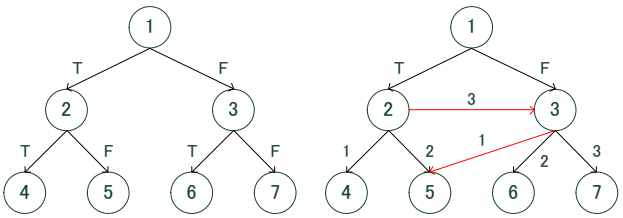


図 3 パスが独立の場合(左)と独立でない場合(右)の例

表 1 GA の実行パラメータ

| | 独立なパス | | 独立でないパス |
|---|---|---|---|
| | 提案手法 | 単純変異 | |
| 個体数 | 1000 | 1000 | 800 |
| 遺伝子長 | 80 | | 500 |
| 交叉率 | 0.9 | 0.9 | 0.9 |
| 突然変異率 | 0.015 | 0.02 | 0.01 |
| テストケース | $t_i(x,y,z)$ | | $t_i(x,y,z,a,b,c)$ |
| 選択方式 | トーナメント選択 | | |
| 交叉方式 | 一様交叉 | | |
| パスの総数 | 80 | | 不明 |
| 変数の範囲 | $0 < \|inputs\| < 200$ | | |
| パス通過条件 | $\|z\| \neq 0, \|z\| < \|xy\|$ | | $10 < \|inputs\| < 100$ |

### 3.2 実験結果

#### 3.2.1 独立なパス

実験結果を表 2 に示す．実験は 50 回繰り返し，解を得るまでに要した世代と時間，パスの網羅率を計測した．

表 2 世代数，実行時間，パスの網羅率

| | 世代数 | 平均時間 | 最長時間 | 網羅率 |
|---|---|---|---|---|
| 提案手法 | 149.3 | 3.98 秒 | 8.92 秒 | 100.0% |
| 単純変異 | 219.14 | 6.05 秒 | 47.33 秒 | 100.0% |

両手法共にパス網羅率 100%のテストパターン生成に成功した．また，処理時間において提案手法の方が性能の高い生成を行うことができていることが確認できる．

#### 3.2.2 独立でないパス

実験結果を表 3 に示す．実験は 20 回繰り返し，解を得るまでに要した世代と時間，パスの網羅率を計測した．世代と実行時間は解を発見したときのものであり，その後一定の世代数を繰り返す処理の世代数と時間を含まない．

表 3 世代数，実行時間，パスの網羅率

| 世代数 | 平均時間 | 最長時間 | パス網羅率 |
|---|---|---|---|
| 244.2 | 52.33 秒 | 75.43 秒 | 99.97% |

評価値は 20 回のうち 19 回が 160，1 回が 159 となった．パスの数を数えると，160 のパスが存在しておりほぼ 100%の網羅率を達成した．

## 4. 考察

3.2.1 の実験ではパスの網羅率 100%のテストパターンの生成に成功した．3.2.2 の実験でもパスの網羅率がほぼ 100%のテストパターンを生成することに成功した．また，従来例として前述した Pairwise 法と Vibration Method との比較では，同じテストデータが入手不能であったため正確な比較はできないが，前者のパス網羅率は 53%，後者のパス網羅率は 92%という結果が報告されている．以上のことから本手法によるテストパターン生成は十分有効である可能性があると考えられる．

一方，今回の手法でテストパターン生成を行うと，テストパターンを生成した時点でプログラムのテストは完了してしているが，生成したテストパターンはテストを行ったプログラムのメンテナンス，バージョンアップ後に再テストを行う際のテストパターンとして有効に用いることができると考えられる．

## 5. おわりに

本稿では，形式仕様に基づいたソフトウェア開発において，GA を用いてプログラムの全てのパスを通過するテストパターンを自動生成する方法を提案した．また，テストケースの集合を染色体として定義し，多次元ベクトルを用いた突然変異を提案した．パス総数が既知のプログラムと未知のプログラムを用いて評価実験を行い，いずれの場合もパスの網羅率がほぼ 100%の精度のテストパターンを生成することを示し．正確な比較ではないが，従来手法よりもパス網羅率を大きく向上できる可能性を示した．

## 参考文献

1) Liu, S. Nakajima, S.: A “Vibration” Method for Automatically Generating Test Case Based on Formal Specifications, In 18th IEEE Asia-Pacific Software Engineering Conference, pp. 73-80 (2011).
2) Pargas, P. R. Harrold. R, R, Peck : Test-Data Generation Using Genetic Algorithms, Journal of Software Testing, Verification and Reliability vol. 9, Issue 4, pp.263-282 (1999).