C プログラムテストデータ検証システムの提案

A Test Data Verification System for C Programs

川喜多　誠，　立岩　佑一郎，　山本　大介，　高橋　直久，
Makoto KAWAKIT[1]，　Yuichiro TATEIWA，　Daisuke YAMAMOTO，　Naohisa TAKAHASHI
名古屋工業大学
Nagoya Institute of Technology
Email: chj15045@stn.nitech.ac.jp, tateiwa@nitech.ac.jp, yamamoto.daisuke@nitech.ac.jp, naohisa@nitech.ac.jp

あらまし：CAPES は，ブラックボックステストにより答案の正誤を判定する．そのための入出力データを正解例プログラムとテストデータ集合から作成する．このとき，正解例プログラム内に一度も実行されない部分（不実行部分）が存在するとテストデータ集合が不足しているためブラックボックステストに十分な入力データが与えられていない疑いが出る．そのため，本稿ではプログラムの静的，動的解析によりテストデータの網羅率と不実行部分を求めて，提示するシステムを提案し，実現法を述べる．
キーワード：網羅率，テストデータ検証，ソースコード解析，プログラムテスト

1.　はじめに

プログラミング演習システム CAPES(1)は，プログラムの提出に対して自動正誤判定する．この自動正誤判定は出題者が与える正解例プログラムとテストデータ集合によって作成される入出力データの対（正誤判定データ）によって行われる（図１）．このとき，正解例プログラムに一度も実行されない部分（不実行部分）が存在すると出題者の意図に沿った正誤判定が十分になされていない可能性がある．
このため，正解例プログラムに不実行部分がないか調べて，もしある場合にはテストデータを追加してテストデータ集合を変更する必要がある．このとき，プログラムが複雑になると，次のような問題が生じる．

問題点1.　プログラムを追いかけて，不実行部分がないかを調べるのは困難になる．

問題点2.　不実行部分が無くなるようにテストデータ集合を変更するのが困難になる．

本稿では，以下のように上記問題の解決を図る．

（１）　テストデータ集合の命令網羅率(2)，分岐網羅率(2)を求めることで不実行部分の検証を行う．

（２）　正解例プログラムの実行されない命令，分岐のソースコード中での位置が分かるように可視化する．

（３）　プログラムの実行時の処理の流れが分かる表現として制御フローグラフ(2)を用いる．

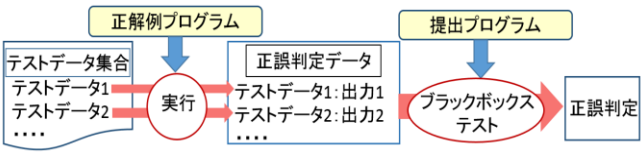

図 1 正誤判定の構成図

2.　提案システム

上記の方針に従い，以下の機能を有し，図 2 の構成のテストデータ検証システムを提案する．

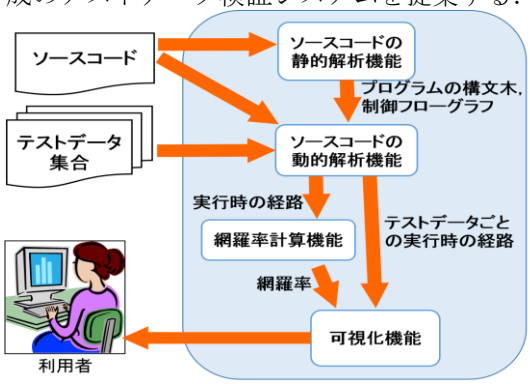

図 2 提案システムの構成図

機能1. ソースコードの静的解析機能：ソースコードを字句解析，構文解析することで構文木を取得し，構文木から制御フローグラフを作成する．

機能2. ソースコードの動的解析機能：構文木を使い，命令，分岐，関数呼出の実行時に出力するように変換したソースコードをテストデータ集合に基づいて実行し，実行時経路を求める．

機能3. 網羅率計算機能：全ての命令，分岐を取得し，また，実行時の経路から実行される命令，分岐を取得し，命令網羅率，分岐網羅率，テストデータ集合の組み合わせ命令網羅率，分岐網羅率を計算する．

機能4. 可視化機能：ソースコードの解析結果や網羅率を可視化する．実行時の経路や網羅率を取得し，不実行命令，分岐をソースコード中で強調表示などにより提示する．また，Dot ファイルを生成し，graphviz(3)を使用し，画像ファイルに変換することで制御フローグラフを提示する．

3. ソースコードの動的解析の実現法

ソースコードの静的解析機能から取得した構文木より命令，制御構文，関数定義の情報を取得し，図 3 のように，それぞれの実行の直前に出力が行われるように出力文を加えたソースコードへと変換する．変換したソースコードを，テストデータ集合を入力とし，実行する．その結果，正解例プログラムの出力とともに，命令，制御構文，関数呼出が実行順に出力される．この出力内容から命令，制御構文，関数呼び出しの関するものだけを抜き出すことで実行列を取得する．実行列を参照し，制御フローグラフを走破する．走破する際に通過したノードとアークにフラグを立てることでフラグの有無で通過したノードとアークが分かり，命令，分岐の実行の有無を取得することができる実行時の経路を生成する．

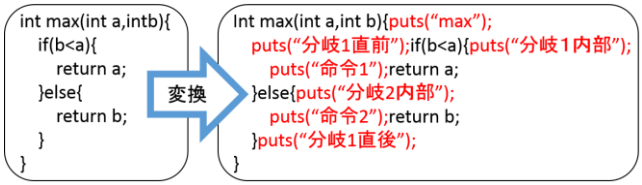

図 3 ソースコードの変換

4. テストデータの検証例

JAVA を用いて，提案システムのプロトタイプシステムを開発した．プロトタイプシステムの出力画面を用いて，テストデータを検証する流れを説明する．

1. ユーザによりプログラムとテストデータが与えられると，網羅率と不実行命令を求めて提示する（図 4）
2. 制御フローグラフを求めて提示する．（図 5）
3. テストデータを追加されると，組み合わせ網羅率と不実行命令を求めて，提示する（図 6）
4. 命令網羅率が 100%である場合，検証終了する．命令網羅率が 100%でない場合は，命令網羅率が 100%になるまでテストデータを追加していく．

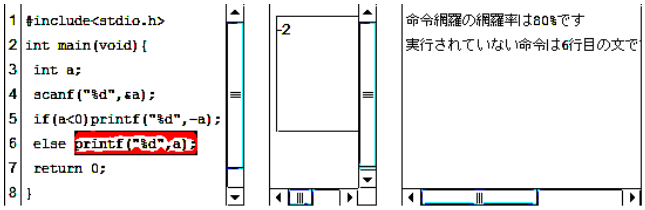

図 4 プロトタイプシステムの出力画面の例 1

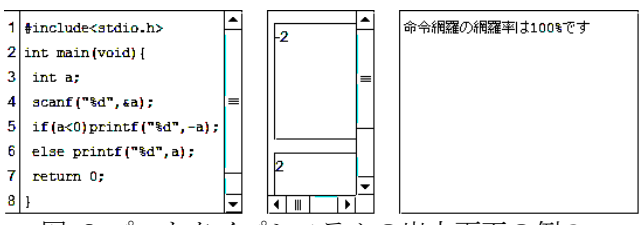

図 6 プロトタイプシステムの出力画面の例 2

5. 評価実験

網羅率を向上させる作業に解析結果と網羅率の提示が与える影響の評価実験を行った．この結果，複雑なプログラムの場合に，何も提示しない場合に比べて，これらを提示すると，より網羅率が向上することが分かった．同時に，作業時間が増大することが確認され，不実行部分を解消するテストデータを探す助けとなるような解析がさらに必要であることが分かった．また，アンケートにおいて，提案システムに追加してほしい機能として，実行されない命令に到達する実行経路中に通過する条件分岐の条件式の提示が挙げられた．

6. おわりに

本稿では，テストデータ集合を命令網羅率，分岐網羅率を指標として検証するシステムを提案し，実現方法を示した．

今後の課題としては命令網羅率や分岐網羅率をあげるための入力の条件を導く方法の実現法が考えられる．

参考文献

(1) 中島，高橋，細川，”プログラミング学習のための QA サイクル-受講者の習得度に応じた問題自動提示メカニズム”，信学論，VOL.J88-D-1，NO.2，pp.439-450（2005）
(2) 高橋，丸山，”11.2 節 テスト技法”，ソフトウェア工学，森北出版株式会社，pp.145-150（2010）
(3) Graphviz – Open source drawing software http://www.research.att.com/sw/tools/graphviz/

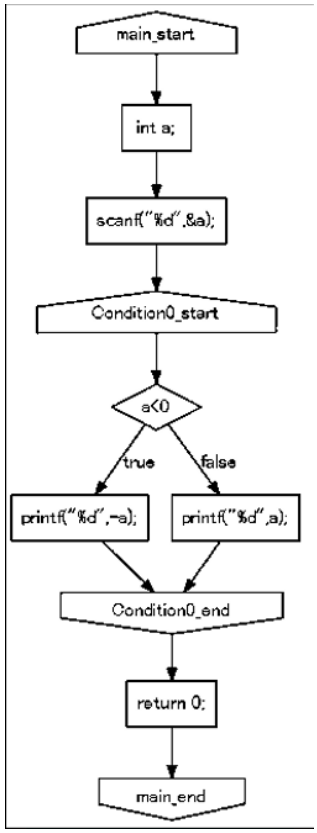

図 5 制御フローグラフの描画例