機能化によるプログラム記号実行法*

2Q−06

森本 紘史†　佐藤 匡正‡
島根大学大学院 総合理工学研究科§

1.序論

プログラムに対して試験する際に検出すべきプログラム誤りは,プログラム上の論理ミス,内部設計における仕様違反,要求分析および外部設計の誤りが挙げられる.ここで,論理ミスおよび仕様違反を記号実行法[1]により検出する方法を考える.記号実行は,入力プログラムの実行経路を表わすパス条件および入力記号値を含む代数式を出力する.この事は 論理ミスおよび仕様違反を検出する上で有効である.

記号実行法は,実際の値ではなく記号を入力として実行する静的なプログラム解析法である.入力されたプログラムが通り得る全てのパスを生成するために,繰り返しに対する処理において致命的な問題が起こる.このため,記号実行法の研究は近年行われていない.本稿では,記号実行法における問題点の改善策として機能化を提案し,その正当性の確認および実行結果の分析を行う.

2.機能化による記号実行

2.1 問題点の整理

2.1.1 繰り返しの展開不可

繰り返しの終了条件に動的なデータを含む時,終了条件を解釈できず繰り返しの展開が不可能となる場合がある.

2.1.2 パス数の増大

繰り返し内のパスを全て展開した結果,パス数が膨大となり仕様との追従性確認に支障を来す.

2.2 機能化

この問題点の改善策として,機能化を提案する.

機能化とは,プログラムから繰り返しの機能を関数として抽出する方法である.

2.2.1 アルゴリズム

関数毎に区切り以下の処理を行う.

(1)繰り返しの関数化[2]

①関数範囲の確定

繰り返し構造内の変数が最後に設定されている処理式および繰り返し構造を関数範囲とする.

②引数値および関数値

確定した関数範囲内の変数で,関数範囲以前に設定される変数を引数値として定義する.関数値は,関数範囲内の変数で関数範囲以後に再設定される前に参照される変数をとする.

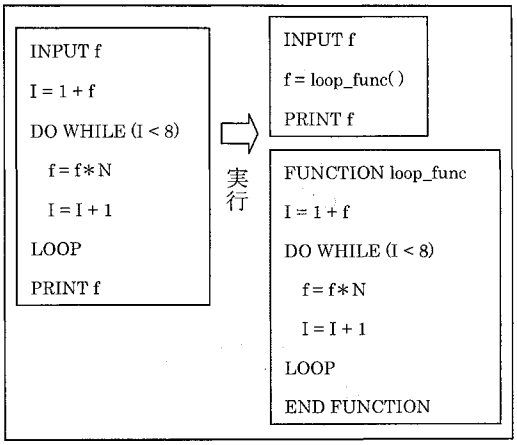
(2)関数内記号実行法

関数範囲内にある処理列に対して記号実行を行い,関数値および引数値の選定を行う.

(3)無効条件の削除

選択構造の組合わせにより論理的に矛盾のある実行経路が生じる事がある.これを無効なパスとして削除する.パス上の変数の値について設定・参照関係に着目し無効なパスを取り除く.

表 1：機能化による記号実行例(使用言語：BASIC)



*Symbolic Execution Reduced Paths Through Functional partition

†Hiroshi Morimoto, ‡Tadamasa Satou
§Faculty of Science and Engineering, Shimane

2.2.2 アルゴリズムの効果

繰り返しの関数化により内部の処理を独立させる事ができパス数を削減でき,無効条件の削除によりプログラムの機能において無駄なパスを削減できる.

2.2.3 例示

表1に実行例を示す.

2.3 表記法

実行結果の視認性を考え頻繁に使用される組み込み関数の略記や長い変数名や関数名は置き換えを行う.これにより機能の理解が視覚的に容易となる.

2.4 関数関連の図示

各関数の関連を図示する支援ツール[3]を組み合わせて用いる事により関数単体だけでなく,関数同士の関連をも可能である.

2.5 制約条件

当面は静的な解析を対象とし，動的データ構造は考えない.また, GOTO文を持つプログラムは対象外とする.

3.正当性と分析

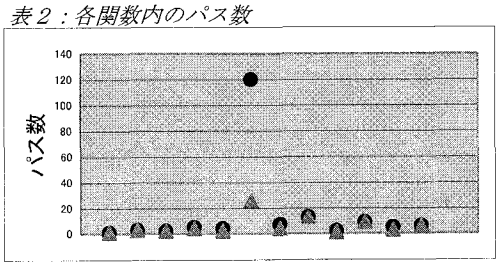
3.1 アルゴリズムの正当性

8〜494 STEPのサンプルプログラム15個に対して実行した結果,アルゴリズム上の誤りは発見されなかった.

3.2 パスの削減状況

表2はあるサンプルプログラム内にある各関数のパス数(▲)および機能化処理後のパス数(●)を示したものである.

表2：各関数内のパス数

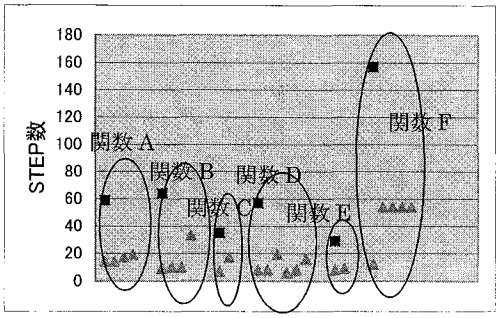


パス数が少ない場合は,関数のパス数の削減はほとんど行われないが,パス数が 120 あった場合には,機能化により26に削減されている.これは，機能化処理によって制御構造の組み合わせによって関数内の総パス数が繰り返し構造内のパス数が積で計算されていたのが機能化された関数間の和の計算量になった事に起因する.

3.3 各関数のSTEP数

表3に実行前の関数における STEP 数(■)および機能化を行った関数の STEP 数(▲)を示した.対象となった関数A〜Dの区分は楕円で囲んで示す.機能化処理後は 69.6%の関数が 8〜20STEPの関数に区切られていた.この事から,機能化処理によって関数のSTEP数がほぼ同程度のSTEP数の関数に分割されたと言える.

表3：関数毎のパス数



4.結論

記号実行法を用いてプログラムから仕様成分の抽出を行う方法を提案し,そのアルゴリズムの正当性を確認した.この技法によってパス数を削減する事ができ,ほぼ一様な STEP 数でプログラムを区切れた.この事はソフトウエアのリエンジニアリングという面で応用できる可能性がある.

参考文献

[1]J.C.KING, "Symbolic Execution and Prog-ram Testing, "CACM, Vol.19, No.7July 1976, pp. 385-394""

[2]森本, 佐藤 : 記号実行における繰り返しの関数化, 平成 13 年度電気・情報関連学会中国支部連合大会

[3]岸本, 佐藤 : プログラムにおける関数表示の有効性, 平成 12 年度電気・情報関連学会中国支部連合大会