5A-02

ソフトウェア品質向上を目指す高カバレッジ単体テスト設計支援ツールの改良

斎藤　正己†　　鎌田　典彦†

日本電気通信システム株式会社†

1．はじめに

ソフトウェアの開発では、自らが開発したソフトウェアに対してテストを十分に実施する必要がある。しかしソフトウェアによっては、十分なテストのためのテスト項目作成が難しい場合がある。そこで我々は単体テストに着目し、高カバレッジな単体テストの設計支援を行うツール[1]を開発した。

本ツールを小規模なソフトウェアの開発プロジェクトで試行すると、「ソースコードのレビュー工数を削減できた」、「単体テスト項目漏れチェック作業の工数を削減できた」との結果を得た[1]。そこで大規模なソフトウェアを開発するプロジェクトへ導入を試みたが、ツールの解析時間が長くなり、テスト計画通りに作業出来なかった。そしてユーザへのインタビューを通じて、テスト計画立案のために解析時間を事前に予測できる必要があると判明した。

本稿では、単体テスト設計支援ツールにおける解析時間の予測方法の検討について報告する。

2．ツールの概要

本ツールはソースコードから MC/DC[2][3]を満たすテスト項目を関数単位で自動生成し、テスト項目ごとに真理値表(図 1)、ソースコード塗り分け図(図 2)、制御フロー図(図 3)を出力する。

真理値表はテスト項目実施のためにソースコード内の分岐の各条件の true/false 設定を示す。ソースコード塗り分け図はテスト項目実施時に動作する部分を赤字で示す。制御フロー図はフロー図上にて当該項目で動作する部分を赤い実線で示す。

テスト実行条件:1
判定式: (Tdata.type < 20) && (Tdata.shape < 20)
(行番号:2)

Tdata.type < 20	Tdata.shape < 20	判定
true	true	TRUE
・・・		

図 1　真理値表

Improvement of Unit Test Designing Tool of Wide Code Coverage for Developing High-Quality Software
†Masami Saito, Norihiko Kamata
†NEC Communication Systems, Ltd.

```
1   int Func_DAT() {
2    if ((Tdata.type < 20) && (Tdata.shape  < 20)) {
3      Tdata.price = Tdata.price * 2;
4    } else if (Tdata.price < 1000) {
                ・・・
```

図 2　ソースコード塗り分け図

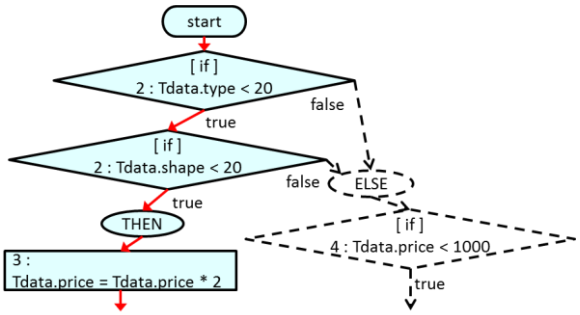

図 3　制御フロー図

3．ツールに対する要望

大規模なソフトウェアの開発プロジェクトで本ツールを試行すると、以下の要望が挙がった。

要望１．解析時間の長い関数があり、無駄な待ち時間が発生する。単体テスト作業の計画立案のために、関数単位での解析時間を概算でよいので予測可能にしてほしい。

要望２．解析時間の長い関数はマシン環境のメモリをかなり消費する。よってツール実行時に解析時間が一定時間を超えた場合、またはメモリ消費量が一定量を超えた場合は、その関数の解析を中止したい。（これらの関数はマシンに負荷のかかっていない時に再度解析する。）

ユーザへのインタビューを通じて、上記の要望を満たすことが必要なのが分かったので、関数単位で解析時間を予測する方法を検討する。

4．サイクロマティック複雑度の活用

本ツールの動作論理を整理すると、まず制御フローグラフをもとに全てのテストパターンを求めて、そこから MC/DC を満たすテストパターンを抽出する。初期調査では、制御フローグラフからテストパターンを求める処理に時間がかかることが判明している。

そこでソースコードのメトリックス値の一つである「サイクロマティック複雑度」に着目する。サイクロマティック複雑度はソフトウェアの複雑度を示す指標であり、制御フローグラフより求められる。よって、本ツールの処理時間とサイクロマティック複雑度の間には何らかの因果関係があると推測できる。

５．測定

解析時間の目安を求めるため、実プロジェクトのソースコードから関数 435 個を抽出して本ツールでの解析時間を測定する

５．１　制限条件

今回の測定ではテスト項目を得ることは必須でないため、次の制限値を設定して、制限値を超える場合は、テスト項目出力を待たずに解析を強制終了する。

(1)時間制限：１関数の解析時間の上限は 1000 分(約 16 時間)

(2)メモリ制限：１関数の解析で使用可能なメモリ上限は物理メモリ及び仮想メモリ共に 5GByte

５．１　測定結果

5.1 節の制限下で測定した場合の、サイクロマティック複雑度と解析結果の関係を表 1 に示す。サイクロマティック複雑度は昇順に均等幅で分割し、Level.1～Level.4 の４段階に分類した。

Level.1 は全ての関数を解析完了し、Level.2 は解析中止が 1 個のみとなり、Leve.3 は解析完了数と解析中止数が同数となり、Level.4 は関数が 1 個のみで解析中止という結果だった。全体的な傾向としては、一部の特異な測定値を除くと、サイクロマティック複雑度が小さいほうが解析完了できる確率が高いことが分かる。

表１　サイクロマティック複雑度と解析結果

	解析完了数	解析中止数	解析完了率
Level.1	381	0	100%
Level.2	42	1	98%
Level.3	5	5	50%
Level.4	0	1	0%
合計	428	7	98%

次に、解析完了した関数におけるサイクロマティック複雑度と解析時間の関係を表 2 に示す（時間の単位は分）。Level.1 は殆どが 1 分以下である。Level.2 も大部分が 9 分以下であるが、1 時間を超えるものもある。Level.3 は測定値が少ないが、9 分以下のものが半数以上を占める。全体的な傾向としては、一部の特異な測定値を除くと、大部分が 1 分以下であり、サイクロマティック複雑度が大きくなると解析時間の長いものが出現することが分かる。

表２　サイクロマティック複雑度と解析時間

時間（分）	0～1	1～9	～19	～39	～59	60～
Lev.1	380	1	0	0	0	0
Lev.2	25	10	4	1	0	2
Lev.3	1	2	0	1	0	1
Lev.4	0	0	0	0	0	0
合計	406	13	4	2	0	3

６．考察

5.1 節より、特異な測定値は存在するものの、全体的にはサイクロマティック複雑度が小さいと解析時間も短いことが分かる。また今回の測定結果は、おおよその解析時間を予測する目安になるので、3 節要望 1 を満たすことができる。

更に、解析時間の予測値を外れる可能性を考慮し、5.1 節で述べた時間制限とメモリ制限をツール自身が意識して、上限値を超えた場合は解析中止する機能を追加する。この機能により、3 節要望 2 を満たすことができる。

７．まとめ

サイクロマティック複雑度と本ツールの解析処理の関係性に着目し、サイクロマティック複雑度からおおよその解析時間を予測できることが分かった。なお、全体の傾向を外れる特異点も存在することから、他の要素も解析結果に影響する場合があると想定して原因調査する予定である。そして、より精度の高い解析時間の予測を目指す。

また、今回の結果をもとに大規模ソフトウェアの開発プロジェクトへの導入を推進し、併せて本ツール活用による定量的効果の測定も行う。

参考文献

[1]斎藤他,"高カバレッジ単体テスト設計支援ツールの活用によるソフトウェア品質向上手法の提案",第 13 回情報科学技術フォーラム論文集(2014)

[2] Kelly J. Hayhurst, Dan S. Veerhusen, John J. Chilenski, and Leanna K. Rierson, "A Practical Tutorial on Modified Condition / Decision Coverage", NASA/TM-2001-210876 (2001).

[3] Thomas K.Ferrell, Uma D.Ferrell, "RTCA DO-178B/EUROCAE ED-12B", The Avionics Handbook, SECTION IV Software, 27 (2001)

[4]高橋寿一,"知識ゼロから学ぶソフトウェアテスト"(2005)