1983.4

S.13-3 最近のマイクロコン・ソフトウェアのテスト技法

中所武司（日立製作所システム開発研究所）

1. まえがき

ソフトウェアの生産性と信頼性の向上は、最近のマイクロコンピュータをはじめとする計算機応用分野の広がりに伴い、ますます重要な課題となってきている。なかでもテスト工程は、ソフトウェア開発費用の半分を占め、生産性向上に重要であるばかりでなく、品質保証に不可欠の重要工程である。[1] しかしながら、テストデータを用いてプログラムを実行し、その結果を調べるというプログラムの検証方法は、"exhaustive testing"とも呼ばれ、実用面で次のような問題がある。

(1) 効果的なテストデータの効率的作成方法がない。

(2) テストの準備や結果の確認作業に手間取る。

これらのテストの問題は、最近の16ビットマイクロコンピュータの普及に伴うソフトウェアの大規模化によって、この分野でも重要になってきた。そこで、本文では、まずテストデータ作成法やテスト実行支援を中心とした、現状での実用的なテスト技法について述べ、後半でマイクロコンピュータソフトウェア用として実用化されているテストツールについて述べる。

2. ソフトウェアテスト技法

プログラムのテスト技法は大まかには静的テストと動的テストに分けられる。

2.1 静的テスト

これは、プログラムを実行することなく、ソースプログラムを解析して誤りを調べる方法で、人手で行うものとして机上テストがあり、プログラム作成者以外の人が加わって行う場合は、コードレビューとかコード検査[2]と呼ばれる。静的テストの自動化ツールとしては、変数の値の定義と参照に関するデータフローを解析して、未定義変数を参照する命令などを検出するものがあるが、このようなプログラム解析ツールによって自動的に検出される誤りは限られている。

2.2 動的テスト

これは実際にプログラムを実行して、その動作の誤りを調べる方法で、最も一般的に行われている。この動的テストには次の2つの作業が必要である。

(1) テストデータと予想結果の作成

(2) テスト実行（テスト環境設定、プログラム実行、結果確認、デバッグ）

現在の動的テストによる検証方法では、効果的なテストデータセットを選ぶことと、テスト実行を効率的に行うことが重要である。そこで、これらについては各々、3章と4章で詳しく述べる。

3. テスト項目／テストデータ作成法

ほとんどのプログラムでは考えうる入力データの数が膨大になるため、そのすべてを試すことは不可能である。従って、限られた時間と費用の中で高い品質保証の与えられるテストデータのセットを作成する必要がある。その代表的な方法として、機能仕様から作成する機能テストとプログラムの構造に基づく構造テスト、および両者を混合した領域法などがある。

3.1 機能テスト

これはブラックボックステスト、仕様テストとも呼ばれ、プログラムの機能仕様からテストケースを作成するものであるが、具体的技法は少ない。人手による一般的手法としては同値分割法[3]がある。これは機能仕様に記述された入力や出力に関する外部条件を細かく数えあげ、各々について有効な入力条件や出力条件、および無効な入力条件を表に記入していく。そして、この表に基づいてテストデータを作成するが、その時、有効条件については1つのテストデータが多くの条件を含むように、また無効条件については個別に作成する。さらに、具体的な値としてはその条件の限界値や最小単位分だけずれたものを選ぶ。一方、ツール化されている技法としては、機能仕様を組合せ論理で表現し、テストケースを自動生成する原因結果グラフ法がある。その詳細は5.1節で実用ツールに従って述べる。

3.2 構造テスト

これは、ホワイトボックステストとも呼ばれ、プログラムの制御構造[3]を有向グラフ化して、そのパス解析に基づいてテストケースを選ぶ。その時テスト網ら基準を用いるが、通常のプログラムはパスの数が膨大で、全パス実行は不可能であるため、制御フローグラフのすべてのノード（即ち、すべての文）を網らするC。やすべてのアーク（即ち、すべての分岐）[4]を網らするC1などの簡易尺度が用いられる。C1の場合のテストデータ選択手順は次の通りである。

(1) すべての分岐がいずれかに含まれるように、パスの最小セットを選ぶ。
(2) 各パスを通過するためのパス条件を求める。
(3) 各パス条件を満たす入力データ値を求める。

例えば、図1のプログラムを考える。2つの条件分岐の真と偽の場合の分岐をT1, F1およびT2, F2とすると、T1, F2, F1, T2の順に実行されるパスを選べば全ての分岐が網らされる。入力データ列としては（100，0）を選べば良い。なお、網ら基準がC。の場合は、T1，T2の順に実行されるパスで良いので、入力データは（101）で良い。

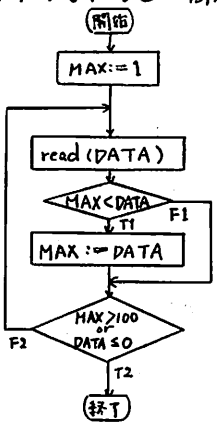

図1 プログラム例

このような構造テストには次のような問題がある。

(P1) 実行不可能なパスの選択

手順(1)でパスを機械的に選んだ場合、パス条件が常に偽となる実行不可能なものが混じる。そこで、実用化されている構造テスト支援ツールは、用意したテストデータをすべて実行した時に、未実行となる文や分岐を検出するものが多く、それを通過するパスの選択は人手に任される。

(P2) 大規模ソフトウェアにおける完全網らの困難さ

システム全体のテストでは網ら性の100％達成はコスト的に難しい。その場合は全体を幾つかのサブシステムに分割し、各々の結合テストで100％を達成するようにする。

(P3) 全パス網ら基準との格差

全分岐網らと全パス網らの間の大きな隔たりを埋めるため、次のような基準が提案されている。

(1) 繰返し処理に対しては、その繰返し条件の成立回数が0回，1回，最大数の場合を網らする。
(2) 繰返し処理部分以外は、全パスを網らする。
(3) データフロー解析に基づき、各変数の値の定義と参照のすべての組合せを網らする。[5]
(4) 連続して実行されうる個所の分岐の組合せをすべて網らする。等々。

(P4) 分岐条件自身のテスト不十分性

パス選択を基本とした構造テストでは、分岐条件の成否の方に注目するため、分岐条件自身の誤りが見逃されやすい。そこで、複数の条件式からなる場合は各々の条件の成否を含むと共に、条件が比較演算式の時は演算子に関係なく、＞，＝，＜の3種類を含むようにテストデータを選ぶ。

(P5) 欠陥パスの検出不可

構造テストでは必要な機能（従って必要なパス）がインプリメントされていないという誤りは検出不可能であり、機能テストが不可欠である。

(P6) 品質評価基準としての過大評価傾向

全パス網ら基準ではテストデータ数に対するテスト率が線形特性を持つが、全分岐網ら基準では凸曲線になり、100％未満の所でテスト率が過大評価される。そこで、著者ら[6]はある分岐の実行に伴って必ず実行される他の分岐をテスト率測定の対象外とする方式により、この欠点を改善している。

3.3 領域法

これは、入力領域を同じ結果を導く部分領域に分割して、各々からテストデータを選択する方法で、領域分割を3.1節の同値分割法で行う場合、パス解析に基づく場合、双方を併用する場合がある。実用的見地での汎用性は他の2方式に劣るので詳細は略す。

4. テスト実行支援

テストの準備や結果の確認作業を効率良く行うためには、テストベッドシステムと呼ばれるテスト実行支援システムが有効である。代表的機能としては (1) 単体、結合テストのための環境模擬機能、(2) 入力データと予想結果および環境設定などを記述するテスト手続き言語、(3) 会話型デバッグ機能、(4) 構造テスト支援機能、などが考えられる。（5.3節参照）

5. 実用ツール

5.1 機能テスト支援ツール

3.1節で述べた原因結果グラフ法を支援するツールとして、古川らのAGENT[10](Automated GENeration system for Test cases)がある。本技法は機能仕様を組合せ論理で表すもので次の手順で行う。

(1)仕様を適当に分割し、各々の原因と結果を識別する。
(2)図2のような原因結果グラフを作成する。
(3)これをある規則で決定テーブルに変換する。(図3)
(4)この決定テーブルの各列をテストケースに変換する。

図2で、～、∧、∨は否定、論理積、論理和を表す。Eは複数の原因が互いに排他的であることを示す。このような制約条件は幾つか用意され、余分なテストケースの生成を防いでいる。図3は、図2のグラフを定形式で入力した時のAGENTの出力の一部で、この例では6個のテストケースが自動生成されている。

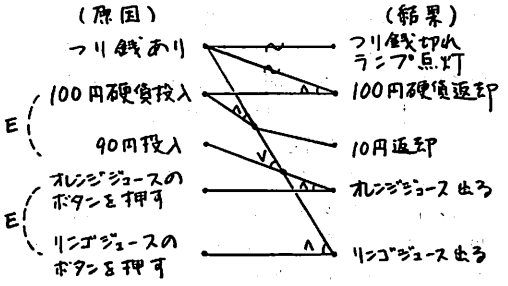

図2 原因結果グラフの例

NO.I	NODE	1	2	3	4	5	6
1	N1	ｱﾗｾﾝ ｱﾘ		X	X		X
2	N2	100ｴﾝ ｺｳｶ ﾄｳﾆｭｳ	X		X		X
3	N3	90ｴﾝ ﾄｳﾆｭｳ				X	
4	N4	ｵﾚﾝｼﾞ" ｼﾞｭｰｽ ﾉ ﾎﾞﾀﾝｵｽ	X	X	X	X	
5	N5	ﾘﾝｺﾞ" ｼﾞｭｰｽ ﾉ ﾎﾞﾀﾝｵｽ					X
8	X1	ｱﾗｾﾝ ｷﾞﾚ ｰ ﾗﾝﾌﾟ ﾃﾝﾄｳ	X		X		X
9	X2	ｺｳｶ ﾍﾝｷｬｸ	X				X
10	X3	10ｴﾝ ﾍﾝｷｬｸ		X	X		
11	X4	ｵﾚﾝｼﾞ" ｼﾞｭｰｽ ﾃﾞﾌﾞ		X	X		
12	X5	ﾘﾝｺﾞ" ｼﾞｭｰｽ ﾃﾞﾌﾞ				X	

図3 AGENTのテストケース出力例

5.2 構造テスト支援ツール

最近、テスト網羅率測定と未実行部の抽出を行うツールが幾つか開発されているが、マイクロコン用としては、遠藤らのCAPT[9](Coverage Analyzer for Program Testing)がある。これはPL/Mプログラムを対象とし、外部ハードウェアトレーサを用いて収集した命令アドレストレースデータからテスト網羅率と未実行部を出力する。図4の出力例ではC0, C1の他にセクションとモジュールの実行率をSS0, MS0で示している。

```
PROGRAMID=TL3SVP
TESTID=T000001
DATE=82-01-08

**COVERAGE INFORMATION**

                    ****************ACCUMU****************
                    SECTION     MODULE     SEGMENT     BLOCK
WHOLE NUMBER              2        244        3202      2894
EXECUTED NUMBER           2         46         277       325
COVERAGE             SS0=100%  MS0= 18%   C1= 8%    C0= 11%
```

図4 CAPTの網羅率出力例

5.3 テスト実行支援システム

ここでは、筆者らが開発した16ビットマイクロコン68000用のテストデバッグ支援システムHITS[11](Highly Interactive Testing-and-debugging System)について述べる。本システムは68000用のアセンブラおよび高級言語Super-PL/Hで記述されたプログラムの機械語オブジェクトを汎用大型機HITACMシリーズ上でシミュレーション実行しながら、効果的かつ効率的なテストとデバッグを行うものである。図5にシステム構成を示す。特に設計上の特徴としては、小規模から大規模までのソフトウェアを対象とし、単体テストからシステムテストまでを支援すること、複数言語のシンボリックテスト支援、系統的テストと効率的デバッグの一元化、バッチ処理と会話型処理の両用化、などがある。主な機能は次のようなものである。

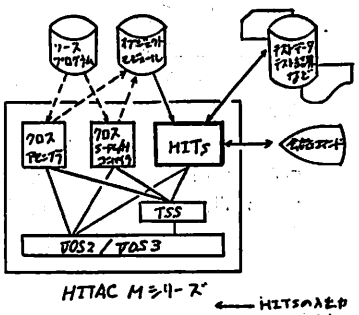

図5 システム構成

(1) テスト手続き記述言語

テストのための環境設定や入力データ、結果照合などはテスト手続きとしてまとめて記述する。その記述言語はコマンド形式とし、ライブラリ化してEXECコマンドで実行できる他、直接端末入力しても良い。

(2) 環境模擬機能

単体/結合テストは環境設定作業が大変なので、その重要性にも拘らずあまり行われてこなかった。そこでHITSでは図6に示すように、未作成の上位、下位モジュールを模擬するドライバ、スタブ定義機能や、外部データ、入出力処理の模擬機能を設けた。

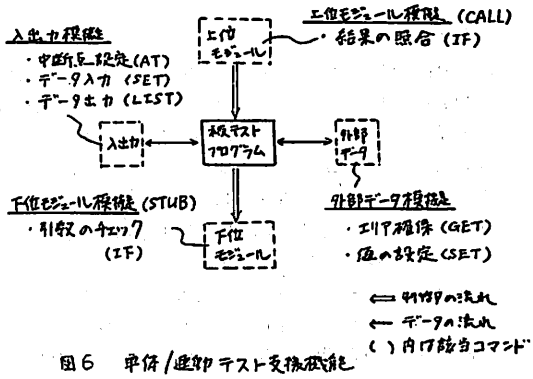

図6 単体/連結テスト支援機能

(3) 環境設定作業削減機能

更に、繰返して用いるコマンド列をマクロ化する機能、入力値だけ変化させて同じテストケースを実行する機能、模擬オブジェクトのリンク機能などを設けて、テスト作業を減じた。

(4) デバッグ機能

誤りを検出したテスト手続きは、一時的なデバッグ用コマンドを付加しながら会話型実行できる。その時、文番号による中断点設定、変数名による値の設定と表示、制御トレースおよびデータトレース（変数値の変化の追跡）などができる。また、レジスタ類の値の設定、表示の他、アドレスエラー等の異常時のダンプ出力や逆トレースを事前に指定できる。

(5) 構造テスト支援

全文網らや全分岐網ら基準に基づくテスト率測定、および未実行の文や分岐の表示を行う。この情報は累積することができる。

図7にHITSのテスト手続きとその会話型デバッグの例を示しておく。

```
PROC TP21                              ┐ 新テストプログラムの
 LINK 7:1000H;9:3000H, DO              │ リンケージ方法指定
  INCLUDE SUB1,SUB2                    │
  END                                  ┘
 GET COMDATA,0A000H
 SET STATE=10110000B
 STUB SUB3, DO                         ┐ 手続き SUB3の
  LIST P                               │ スタブ定義
  IF STATE=0F0H, SET P=40              │
  END                                  ┘
CASE C01
   ⋮
CASE C07                               ┐ 7番目のテストケース
 CALL SUB1(3,V)                        │
 LIST V,STATE                          │
 IF V=0, LIST 'ERROR:V=0'              │
 END CASE                              ┘
   ⋮
END PROC
```

(a) テスト手続き

```
[Ready 状態]
OPTION BREAK                           ┐ テストケースC07の
EXEC TP21(C07)                         ┘ 実行指示
[テストケースC07の開始後手続きSUB1の
 実行開始直後に中断。コマンド待ち。]
QUALIFY SUB1                           ┐ デバッグ用コマンドの
AT 17, LIST STATE                      │ 入力
TRACE DATA(STATE)                      │
RESUME                                 ┘
[最初に STATE の値を変えた 所で
 中断。コマンド待ち。]
STOP PROC
[テスト手続きの強制終了。Ready 状態]
```

(b) 会話型デバッグ

図7 HITSの使用例

5.4 入出力シミュレータ

一般に大型計算機を用いてマイクロコン用ソフトウェアのテストを行う場合、論理的処理を記述したプログラムのテストを徹底して行うことは可能であるが、入出力処理部分のテストは難しい。そこで西野らは68000用の高級言語Super-PL/Hに高水準入出力機能を付加して、それを模擬する入出力シミュレータ[10]を開発している。即ち、米国国防総省の新言語ADAの汎用入出力パッケージとテキスト入出力パッケージのサブセットを入出力ライブラリとして用意し、それを用いたプログラムの入出力処理を模擬する。

5.5 テスト手順

以上に述べたようなテストツールを用いたテスト手順は次のようになる。(1)まず原因結果グラフ法が適用可能なプログラムはAGENTなどを用いてテストケースを生成する。(2)次に同値クラス法でテストケースを選ぶ。(3)これらのテストケースからテストデータを作成し、テスト手続きにまとめる。(4)HITSなどを用いてテスト実行する。(5)HITSやCAPTなどで、未実行の文や分岐が検出された場合は、それを通過するテストデータを追加作成し、テストする。(6)テスト網ら幸がある基準に達するとテストを終了する。

6. おわりに

本文では実用的なテスト技法とツールを中心に述べ、研究段階にあるものは略した。またマイクロコンシステム用の開発支援システムに含まれるシンボリックデバッガー等については他に譲る。

参考文献

1) 中所：ソフトウェアのテスト技法、信学会誌、Vol. 64, No. 5, 549-552 (1981).
2) Myers, G.J.：ソフトウェア・テスト技法、(長尾、松尾訳)、近代科学社 (1980).
3) Miller, E.F.：Program Testing：Art Meets Theory, Computer, Vol. 10, No. 7, 42-51 (1977).
4) Huang, J.C.：Error Detection Through Program Testing, Current Trend in Programming Methodology, Vol. II, Prentice-Hall (1977).
5) Rapps, S. et al.：Data-Flow Analysis Techniques for Test Data Selection, The 6th ICSE, 272-278 (1982).
6) 中所：パステストに本質的な分岐に着目した網ら率尺度の提案、情処学会論文誌、23, 5, 545-552 (1982).
7) 古川他：機能テストのためのテスト項目作成手法について、情処学会 ソフトウェア工学研究会資料 16-2 (1980).
8) 遠藤他：トレースデータに基づくマイコン用プログラムのテストカバレージアナライザ、情処第25回全国大会、485-486 (1982).
9) 中所他：16ビットマイコン68000用クロス型テストデバッグ支援システムHITS、情処学会 マイクロコンピュータ研究会資料 20-3 (1982).
10) 西野他：68000用クロス型コンパイラSuper-PL/Hにおける高水準入出力機能の開発、情処第25回全国大会、1219-1220 (1982).