設計自動化 30－1
（1986. 1. 21）

回路分割を用いた テストデータ生成システム

鈴木和弘、門倉敏夫（早大）、
深沢良彰（相模工大）、長谷川拓己（日本電気）

1. はじめに

ハードウェアの大型化に伴い、テストデータ生成や故障検査に要する時間は膨大なものになってきている。そこで、テストデータ生成に要する時間を短縮するために、乱数パターンをテストデータとして用いる試み[1]が行なわれている。しかし、この場合に、高い故障検出率を達成するためには、多くのテストデータを必要とする。特に、順序回路に対してテストデータ生成を行なう場合には、順序回路中のフィードバックを検査するための一連のデータを与えなければならない。このため、順序回路に対して故障検査を行なう場合や、高い故障検出率が要求される場合には、アルゴリズム的にテストデータ生成を行なう必要がある[2]-[4]。しかし、この場合には、テストデータ生成に多くの時間を必要とする[5]。このため、テストデータ生成時のバックトラックを減らし、処理時間を短縮させるアルゴリズム[6]-[7]が開発されている。

我々は、故障検査の初期段階で故障を検出できるようなテストデータを、より短い時間で生成できることが望ましいと考えた。そして、以下に述べる手法を用いたテストデータ生成システムを考案した。

本システムでは、テストデータ生成の対象とする回路を後述する方法により分割し、分割された個々の回路（以下、セグメントと呼ぶ）に対して順次テストデータ生成を行なっていく。また、過去の故障検査時の統計的な情報や使用者の指定したデータを用いて、最も故障が起こる可能性が高いと思われる箇所を求める。そして、この箇所からテストデータ生成の対象としていく。

以下、本システムの特徴、本システムの構成、優先順位の評価方法、セグメントの生成方法の順で述べる。

2. 本システムの特徴

本システムでは、テストデータ生成の対象とする回路を故障情報を用いてセグメントに分割する。そして、個々のセグメントに対してテストデータを生成する。セグメント化を行なうことにより、一度に扱う回路の規模が小さくなり、個々のセグメントに対するテストデータ生成時間は短縮される。更に、回路全体に対してもテストデータ生成に要する時間を短縮することを期待できる。

しかし、セグメント化を行なった場合には、同一箇所に対して重複してテストデータを生成してしまう場合が存在するために、テストデータの数が多くなってしまう。一般に、アルゴリズム的にテストデータ生成を行なった場合には、故障検査よりテストデータ生成に、多くの時間を必要とする。このため、テストデータの数、つまり、故障検査に要する時間より、テストデータ生成に要する時間を短縮する方がより効果的であると考え、セグメントを用いたテストデータ生成を行なっている。

本システムでは、テストデータ生成の対象とする機能ブロックの各下位ブロックに、優先順位を設定する。そして、この優先順位が高い機能ブロックから、テストデータを生成する。この優先順位は、過去に行なった故障検査によって得られる統計的な故障情報を用いて計算される。本システムは、優先順位が高い機能ブロック程、故障が発生している確率が高いと判断し、優先的にテストデータ生成を行なう。優先順位を考慮したテストデータを用いて故障検査を行なうことにより、検査の初期段階で故障が検出される確率が高くなる。

3. 本システムの構成

本システムの構成図を図１に示す。

使用者は、本システムに対してハードウェア

記述を入力すると共に、制御データにより、テストデータ生成を行なうブロック（以下、被検査ブロックと呼ぶ）を指定する。

本システムでは、階層表現されたハードウェア記述を入力として仮定している。グラフ展開部では、グラフライブラリからの情報を用いて、被検査ブロックをマクロ展開し、グラフ形式に変換する。このグラフには、機能ブロック間の接続情報や、その機能ブロックから呼び出される下位ブロックの情報が含まれている。実際のグラフでは、機能ブロックや素子はノードとして、接続情報はアークとして表現されている。

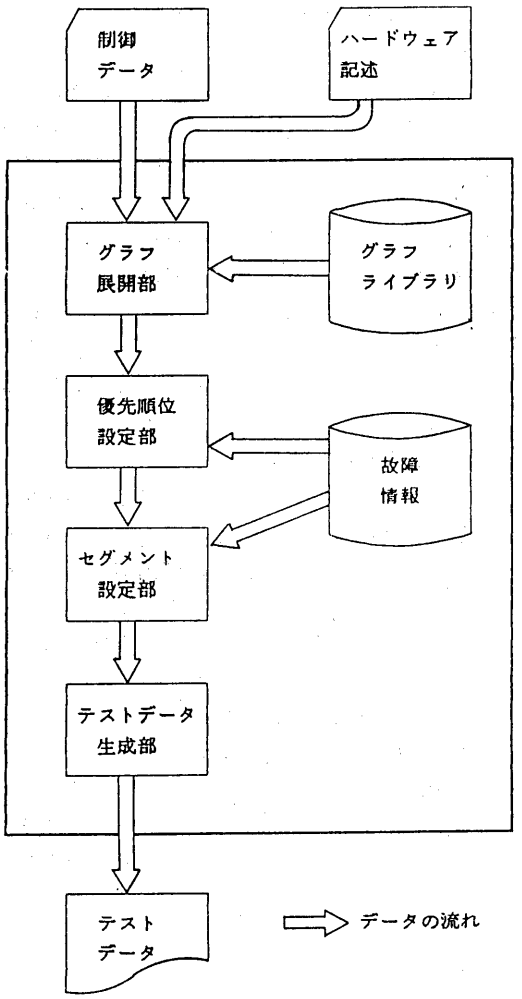


図１．本システムの構成

優先順位設定部では、故障検査の初期の段階で故障を検出するために、被検査ブロックの各下位ブロックに対して優先順位を設定する。そして、優先順位を評価し、故障が発生している確率が高い下位ブロックを求め、このブロックからテストデータ生成の対象としていく。

セグメント設定部では、優先順位が高い下位ブロックに対するセグメントを求める。ここでは、この下位ブロックから入力及び出力方向に対する探索を行ない、故障を検出する可能性が最も高いパス上に存在する下位ブロックをセグメントとしている。

テストデータ生成部では、求められたセグメントに対して、テストデータを生成する。本システムでは、テストデータ生成にD-アルゴリズム[2]を改良して用いている。

優先順位設定部中の処理について、4.の優先順位の評価方法で、また、セグメント設定部中の処理について、5.のセグメント生成方法で述べる。

4.優先順位の評価方法

被検査ブロックに対してテストデータを生成する際、このブロックに複数の下位ブロックが存在する場合には、優先順位を用いて、どの下位ブロックからテストデータ生成の対象とするかを決定する。

テストデータ生成を行なう下位ブロックを優先順位を用いて決定する例を以下に示す。この例で用いる記述を図２に、それに対応する回路を図３に示す。図３において、機能ブロックBLOCK_1の下位ブロックには、FUNC_A、FUNC_Bと、BLOCK_Cが存在している。BLOCK_1をテストデータ生成の対象とする場合、３つの下位ブロックに対する優先順位を評価し、テストデータ生成の対象とする順番を決定する。

ここでは、３つの下位ブロックに対する優先順位が高い順に、

BLOCK_C, FUNC_A, FUNC_B

であると仮定する。この場合には、最も優先順位が高いBLOCK_Cからテストデータ生成の対象とする。

具体的には、

FUNC_A(ARG1, ARG2) and FUNC_B(ARG3, ARG4)の値を真にするための入力データをARG1～ARG4により設定し、BLOCK_Cを活性化する。そして、BLOCK_Cに対するテストデータを生成する。

```
BLOCK_1:
if FUNC_A(ARG1, ARG2)
        and FUNC_B(ARG3, ARG4)
  then BLOCK_C(ARG5, ARG6, ARG7) ;
```

図２．ハードウェア記述例

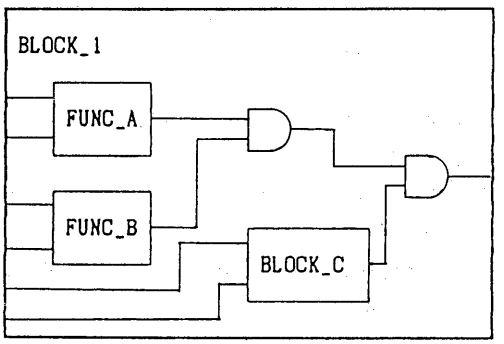


図３．図２に対する回路構成例

次に、優先順位の評価法ついて述べる。優先順位の評価の対象とする項目には、

a)同一の仕様を持つ機能ブロックに対するもの
b)固有の機能ブロックに対するもの
c)機能ブロックの稼働時間

がある。a)は、主に汎用性を持つ機能ブロックに対するものである。そして、同種の機能ブロックを使用しているすべての箇所において、同一の統計情報を参照する。これに対して、b)は、個々のブロックに対して、そのブロックのみの統計情報に基づいて求められる項目である。また、c)の稼働時間を用いてa)、b)の故障率の補正を行なう。これは、ある機能ブロックが、故障を起こす確率は、そのブロックの稼働時間と共に推移すると考えられるからである。優先順位を決定する方法を図４に示す。

```
for all(下位ブロック) do begin
  a)、b)の故障率を計算する;
  c)の稼働時間により各故障率を補正する;
  補正後の各故障率に重み付けをして平均値を
  計算する
end;
平均値が大きい下位ブロック程、より高い優先順位を与える;
```

図４．優先順位決定方法

次に、各項目に対する重みを決定する方法を示す。

・対象としている項目の故障率xiをいくつかのクラスに分割し、それぞれのクラスに属する機能ブロックの数を度数として記録する。
・度数を用いて、故障率の平均値$\bar{x}$、標準偏差σを求める。
・個々の機能ブロックにおけるこの項目に対する重みwiは、

$$wi = (xi - \bar{x}) / 10\sigma + 0.5 \quad \cdots\cdots\cdots (1)$$

この重みwiは、0.5を中心とした偏差値を意味し、各機能ブロックの故障率が、すべての機能ブロックの故障率の分布状況において、どの程度の位置にあるかを示す。

たとえば、ある機能ブロックに対するa)の項目、すなわち、同一の仕様を持つ機能ブロックに対する故障率が、他の機能ブロックより極端に大きい場合、この項目に対する重みwiは大きくなる。これは、その機能ブロックに関して設計上あるいは、製造上の問題が存在すると考えられるためである。

5. セグメントの生成方法

優先順位によって指定されたブロックから、入力及び、出力方向に対して探索を行ない、セグメントを生成する。ここでは、故障を検出する確率が最も高くなるパスをセグメントとしている。以下、テストデータ生成の対象とするブロックの入力をP.I.(Primary Input)、出力をP.O.(Primary Output)とする。

セグメントの生成方法を図５に示す。

```
for all(指定された機能ブロックの出力端子)
  do
  出力端子からP.O.までのパスのうち、
  最も故障検出確率が大きくなるパスを      [a]
  探索する;
for all(パス上に存在する機能ブロック) do
  機能ブロックの入力方向に存在するす
  べての機能ブロックをセグメントに加      [b]
  える;
```

図５．セグメント生成方法

図５の[a]では、優先順位によって指定された機能ブロックから、P.O.までのパスの探索を行なっている。ここでは、最も故障検出確率が大きくなるパス上に存在する機能ブロックをセグメントにしている。これは、指定された機能ブロックの出力データを、いずれかのパスによってP.O.まで伝搬させれば、故障の有無を判断できるからである。

また、[b]では、[a]で指定したセグメントに対してテストデータを生成するために、必要となる機能ブロックをセグメントに加えている。すなわち、セグメント中に存在する機能ブロックに対して、そのブロックより入力方向に存在するすべての機能ブロックをセグメントに加えている。

以下、[a]におけるパスの探索方法について述べる。

6. 出力方向のパスの探索

ここで、パスの探索時に用いる情報の定義を行なう。各機能ブロックに対して、故障情報として以下の情報が与えられている。

- 機能ブロックの故障検出確率(P)
- 機能ブロックを検査する時に必要となるテストデータの数(n)

また、これらの情報から１個のテストデータによって故障を検出できる平均確率(PN)を求める。

PN = (故障検出確率) / (テストデータ数)
= P / n ……(2)

このPNを用いて出力方向のパスの探索を行なう。以下、6.1.で、PNを用いたパスの探索方法について述べる。そして、6.2.で、PNに用いるテストデータ数について評価を行なう。

6.1. PNを用いたパスの探索

6.で定義したPNを用いて、パスの探索を行なう。ここでは、優先順位で指定された機能ブロックから、P.O.までのパスのうち、最もPNが大きくなるパスを求めている。

図６において、優先順位で指定された機能ブロックをBs、Bsから出力方向へのパス上に存在する機能ブロックをB1、B2とする。このときにB1とB2のどちらをセグメントに含ませるかの判断は、

a) B1をセグメントに含ませた場合に、期待されるPN(B1)を求める。
b) B2をセグメントに含ませた場合に、期待されるPN(B2)を求める。

a)とb)を比較し、PNの値が大きい方をセグメントに含ませる。

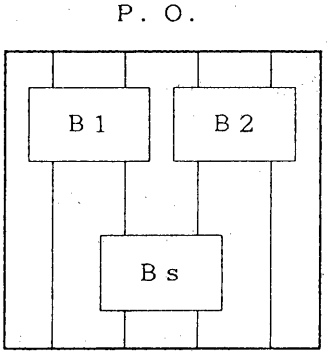


図６．機能ブロック例

以下に、a)および、b)の実際の計算方法を示す。まず、B1をセグメントに含ませた場合は、

PN(B1) = (Ps + P1) / N(Bs , B1) ……(3)

となる。ここで、N(Bs , B1)は、Bs及びB1を同時に検査するために必要となるテストデータの

数を示している。

また、B2をセグメントに含ませた場合は、

$$PN(B2) = (Ps + P2) / N(Bs , B2) \quad \cdots\cdots(4)$$

となる。

以上のPNにおいて、Ps、P1、P2は故障情報として与えられている。しかし、複数の機能ブロックに対するテストデータ数Nは、故障情報として与えられている単一の機能ブロックに対するテストデータ数nを、そのまま用いることはできない。そこで、このNを求める方法について以下に述べる。

6.2.複数ブロックに対するテスト数の評価

複数の機能ブロックに対して、同時にテストデータを生成する場合に、必要となるテストデータ数の評価を行なう。ここでは、評価のパラメータとして、フィードバック数を考える。

回路中にフィードバックが存在している時には、出力データを変更することが困難になってしまう。これは、フィードバックの値を変更しようとしたときには、通常１つ以上のデータを必要とするからである。

また、ある機能ブロックを検査する時に必要となるテストデータの数は、その機能ブロックの入力や出力の接続状況によって大きく変化する。つまり、機能ブロックの入力がすべてP.I.であったならば、必要とするテストデータの数は、故障情報として与えられているｎとなるが、機能ブロックの任意の入力が１や０に固定されていた場合や、入力方向に多くの機能ブロックが接続されていた場合には、検査が不可能もしくは、困難になってしまう。このことから、入力の拘束度を考える。

入力の拘束度　：
　ある機能ブロック入力の値を変更するのに必要なデータの平均数

つまり、この拘束度が大きい入力程、その値を変化させるのは、困難になる。このため、その入力が接続されている機能ブロックに対するテストデータ数Nは、大きくなってしまう。

以下、6.2.1.で、フィードバックの値を変化させるために必要となるデータ数の評価を行なう。そして、6.2.2.で、その結果を用いて機能ブロックを検査するために必要となるテストデータ数を評価する。

6.2.1.フィードバック値の変更

ここで、フィードバックの値を変更するために必要なデータの数について述べる。

フィードバックによって保持される値を状態として考え、回路を状態遷移を用いて表現する。ここで、フィードバックが１箇所の場合の状態を図７及び、図８に示す。この場合は、フィードバックが１箇所なので、状態は、S0及び、S1の２種類、状態の遷移方法は、T0～T3の４通りとなる。また、出力は、フィードバックの値であり、状態S0、S1に等しくなる。

ここで、出力の値を一定値cに設定する場合、すなわち、状態Scに遷移させる場合に必要となるデータの数を考える。cは、0もしくは1である。現状態をSxとすると、次状態はS0、S1の２通りとなる。ここで、次状態がS0となる確率を1/2、また、S1となる確率も1/2と仮定する。

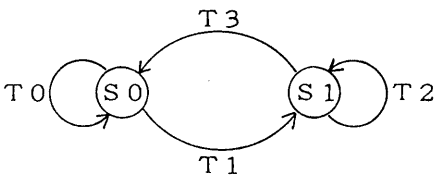


図７．状態遷移図

状態遷移	現状態	次状態	出力
Ｔ０	Ｓ０	Ｓ０	Ｓ０
Ｔ１	Ｓ０	Ｓ１	Ｓ１
Ｔ２	Ｓ１	Ｓ１	Ｓ１
Ｔ３	Ｓ１	Ｓ０	Ｓ０

図８．状態遷移表

まず、１つのデータで値の設定を行なえる確率は、状態Sxから状態Scへ遷移する確率であり、1/2となる。この場合のデータ数の期待値は、

$$1\text{[個]} \times (1/2) \quad \cdots\cdots(5)$$

となる。また、２つのデータで値の設定を行なえる確率は、１つめのデータで値の設定を行なえず、しかも、２つめで設定を行なえる確率である。この場合の確率は、

$$(1-1/2) \times (1/2) = (1/2)^2 \quad \cdots\cdots(6)$$

となり、データ数の期待値は、

$$2\text{[個]} \times (1/2)^2 \quad \cdots\cdots(7)$$

となる。このことより、Ｋ個のデータで値を設定できる場合のデータ数の期待値は、

$$K\text{[個]} \times (1/2)^K \quad \cdots\cdots(8)$$

となる。以上より、フィードバックに一定値を設定するために必要となる平均のデータ数は、

$$1 \times (1/2) + 2 \times (1/2)^2 + \cdots + K \times (1/2)^K + \cdots$$

$$= \sum_{K=1}^{\infty} (K \times (1/2)^K)$$

$$= 2 \quad \cdots\cdots(9)$$

よって、１つのフィードバックに任意の値を設定するためには、平均2個のデータを必要とする。

次に、フィードバックFB1及び、FB2が直列に存在しているパスを考える。FB1の出力にデータを設定するために、平均2個のデータを必要とする。そして、FB1の出力データを2個用いて、FB2の出力データを設定する。これより、FB2の出力データを設定するためには平均、

$$2 \times (\text{FB1.OUT}) = 2 \times 2 = 2^2 \quad \cdots\cdots(10)$$

個のデータが必要となる。

以上のように、パス上にフィードバックがｎ個直列に存在していた場合には、平均2^n個のデータによって出力データを設定できることがわかる。

6.2.2. フィードバック数によるテスト数の評価

6.2.1.において求めたフィードバックの値を変更するためのデータ数を用いて、機能ブロックを検査するために必要となるテストデータ数を評価する。

ここで、図９の回路において、機能ブロックB(i) を検査するために必要となるテストデータ数N(B(i))を評価する。

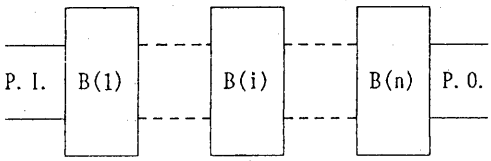


図９．テストデータ数評価例

まず、入力方向に対する処理方法を示す。

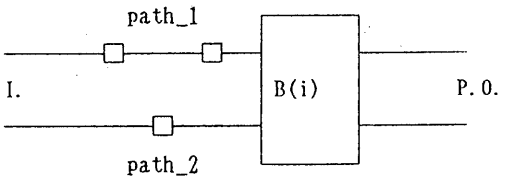
for all(B(i)の入力端子B(i)IN(j)) do
B(i)IN(j) からP.I.までに通過したフィードバック数の最大値をB(i)FBIN(j)とする; ……(11)

$$B(i)FBIN = \max_j (B(i)FBIN(j)); \quad \cdots\cdots(12)$$

図１０．フィードバックの評価方法（入力方向）

(11)において、フィードバックの数が最大であるパスを求めたのは、入力の拘束度を求めるためである。つまり、図１１において、path_1では、フィードバックが２箇所であるのに対し、path_2では、フィードバックが１箇所でP.I.に到達している。この場合の、入力の拘束度は、

path_1により、2^2となる。これは、B(i)の入力の値を変更するのに平均2^2個のデータを必要とすることを意味している。



□：フィードバック

図１１．入力の拘束度

(12)において、最大値を用いたのは、B(i)の入力に対して、最も拘束度が大きい入力端子を見つけるためである。

次に、出力方向に関しては、いずれかのパスを経由し、P. 0. まで出力データを伝搬させればよい。ここでは、最もフィードバックの数が少ないパスを求める。以下に、出力方向に対する処理方法を示す。

```
for all(B(i)の出力B(i)OUT(j)) do
  B(i)OUT(j)からP. 0. までに通過したフィードバック数の最小値をB(i)FBOUT(j)とする;
                                        ……(13)
B(i)FBOUT = min( B(i)FBOUT(j) );
                                        ……(14)
```

図１２．フィードバックの評価方法（出力方向）

(13)及び、(14)で、最小値を用いたのは、出力方向では、最もフィードバック数が少ないパスを求めるためである。

以上、入力及び出力方向に関する処理で求めたB(i)FBIN及び、B(i)FBOUT を用いて、B(i)に対するテストデータ数N(B(i))を計算する。

$$N(B(i)) = n(i) \times 2^{(B(i)FBIN+B(i)FBOUT)} \quad \cdots\cdots(15)$$

上式により、P. I. からB(i) を通り、P. 0. までデータを伝搬させていくときの、テストデータ数を評価できる。

ここまでの結果を用いて、１つのパスを検査する時に必要となるテストデータ数の評価を行なう。

```
for all(パス上の機能ブロックB(i)) do
  B(i)を検査するのに必要なテストデータ数
  N(B(i))を求める;
N(B(i))の最大値を求める;
```

図１３．テストデータ数の評価方法

ここでは、パス上の各機能ブロックB(i)について、N(B(i))を求め、それらの最大値をパスに対するテストデータ数としている。これは、機能ブロックB(k) が最大値を持つと仮定すると、B(k)がこのパスのボトルネックとなり、検査に必要なデータ数を増加させてしまうことを示している。

これより、図６におけるそれぞれの機能ブロック対する平均故障検出確率、(3)及び、(4)は、

$$PN(B1) = (Ps + P1) / \max(N(Bs), N(B1)) \quad \cdots\cdots(3')$$

$$PN(B2) = (Ps + P2) / \max(N(Bs), N(B2)) \quad \cdots\cdots(4')$$

となる。よって、(3')及び、(4')を計算し、より大きい値を持つ機能ブロックをセグメントとする。

以上の処理によって、最も平均故障検出確率が大きくなるパスをセグメントとすることが可能となる。そして、求めたセグメントをテストデータ生成の対象とする。

7. おわりに

本稿では、回路分割を用いたテストデータ生成システムについて述べた。本システムでは、１つのテストデータ当たりの故障検出確率が最も高くなるように回路分割を行なっている。

現在、優先順位の評価、及び、セグメントの生成に関して、本システムのインプリメント及び評価を行なっている。また、現在の課題とし

ては、以下のものがある。

・優先順位を決定する評価関数は、数多くの実例に関して調査を行なう事により、より実用的になるため、様々な回路に対して故障検査を試みる。
・現在の優先順位は、機能ブロックの接続情報のみによって決定されているが、これに、ハードウェアの実装状態に関するデータ、つまり、配線間隔や、配線長なども考慮して、決定を行なう。
・セグメントの生成方法、特に、テストデータ数に対する評価方法を再検討し、より効率が良いセグメントを生成する。

参考文献

[1] R.M.McDermott, "Random Fault Analysis", Proc. of 18th. DA Conf., pp.360-364, (1981).

[2] J.P.Roth, "Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits", IEEE Trans. Computers, Vol. EC-16, No.5, pp.567-580, (1967).

[3] D.R.Schertz, "A New Representation for Faults in Combinational Digital Circuits", IEEE Trans. Computers, Vol. C-21, No.8, pp.858-867, (1972).

[4] J.Rajski, "Fault Influence Function for Identification of Multiple Faults in Combinational Circuits", Proc. of 20th. DA Conf., pp.106-109, (1983).

[5] O.H.Ibarra and S.K.Sahni, "Polynomially Complete Fault Detection Problems", IEEE Trans. Computers, Vol.C-24, No.3, pp.242-249, (1975).

[6] P.Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits", IEEE Trans. Computers, Vol.C-30, No.3, pp.215-222, (1981).

[7] H.Fujiwara and T.Shimono, "On the Acceleration of Test Generation Algorithms", IEEE Trans. Computers, Vol. C-32, No.12, pp.1137-1144, (1983).

[8] 鈴木,門倉,深沢, "統計的な故障率を考慮したテストデータ生成システム", 情処学会第30回全国大会, 7H-2, (1985).