ソフトウェア工学 55－1
（1987. 7. 29）

プログラムテストに用いるパスジェネレータへの一考察

柳沢 隆夫
芝浦 工大

本論文は、プログラムの自動的なテストパス作成に於いて生じる、２つの問題のためのアルゴリズムを考慮している。
これらの問題は、有向グラフの全ての辺を含む最少のパス集合を決定することと、有向グラフの指定された辺の集合を通る最少のパス集合を決定することである。

A study of the Path Generater for Program Testing

Takao Yanagisawa
Shibaura Institute of Technology
Oomiya-shi, Saitama-ken, Japan

In this paper we consider algorithm for tow problems that arise in automatic test path generation for program: the problem of determining a minimal set of paths which contain all the edges of a directed graph and the problem of determining a minimal set of paths which is through a specified edges of a directed graph.

1、はじめに

プログラムテストのための有向サイクルを含まない有向グラフの各々の辺を含む最少パス集合を求める研究は、Deb[1]によって、割当アルゴリズムを用いてグラフをチェインに分割し、輸送アルゴリズムによって、チェインをパス集合に拡張して行われた。その後Ntafos[2]によって、最少流量法を用いて求められている。

一方、有向サイクルが含まれていることを許したグラフに対する最少パス集合の導出についての研究も、これまでに若干行われている[3][4][5]。（Paigeは正則式を用いて導出したが、この方法は簡潔で、計算時間が少なくて解けるが、プログラムが構造化されていないと適用出来ない）。

柳沢は先に、有向サイクルの有無に拘らず、グラフの指定された頂点とパスを通る最少パス集合を求める方法（有向グラフの各節を含む最少パス集合に帰着させて解く）を報告した。しかし、そこでは有向サイクルのコンデンスと強連結成分内のサブパスの導出に、D.F.S法（Depth-First Seach）が用いられている。

本研究は、前回に於ける有向サイクルのコンデンスや、コンデンスされたものを元に戻す手続きが省略されればさらに効率的であることや、最短のパス集合を求めるとさらに有効性が増すという問題を、前回とは異った接近を試み、各有向辺を含み、最少かつ最短なパス集合を求める解法を行った。

この解法は、最少で最短なパス集合を導出する問題と、ネットワーク理論による最少コストフローのS－T単位順枝列フロー分解とが等価であることを示し、続いて、大きなグラフの最少コストフローの導出とS－T単位順枝列フローへの分解に、簡単で効率的なアルゴリズムが応用出来ることを示すことにより行われる。

ところで、プログラムテスト法として、先ずプログラムグラフのテストパスを決定し、次にこれを通る入力データを算出して行うものは、テストパスが不実行パスであったときは破局となる。もし、プログラムテストパスがプログラムのブランチカバーの最少のものであったときは、この問題は重大となる。この問題に対し、Kunduは、全てのブランチをカバーするテスト法として、未だテストされていないパスプレディケイト（predicate、そのパスを実行する入力変数の満足すべき範囲）より、次のテストデータを決定し、この方法を全てのプログラムブランチがカバーされるまで続けるものを提案したが、この方法も、S－Tパス集合が大のときは難解なものとなる。

本研究はテスト法として、先ずプログラムのスタートとディシジョンの直ぐ後にカウンターを挿入し、いくつかの入力データをランダムに入力してテストを行い、次に残りの未だ未テストの部分にテスターを振り向けるテスト法の問題を扱う。

この問題は、プログラムグラフの指定された辺を通るパス集合を求めるものと等価である。本研究は、未テスト辺を含む最少で最短なテストパスをグラフ理論を応用して導出するときに生じた種々の問題を検討した。

2、本研究で使用される主な用語

有向グラフG：空でない集合VとVと素な集合Eと、写像Ψ（：E→V×V）で構成され、Vの元をGの頂点，Eの元を有向辺という。G＝［V,E,Ψ）で表わす。

ネットワークN：有向グラフGと区間重み関数Iと活動f（フロー）とか

ら構成される。Gは節点集合Vと付向枝集合Eより構成される。Iは枝集合上で定義され、fの上，下限を与えるところの閉区間I(e)を与える。fは枝集合上で定義され、フロー公理

$$\sum_{e\in\omega^+(x)} f(e) = \sum_{e\in\omega^-(x)} f(e) \quad (\forall x\in V)$$

を満足する。ここで、$\omega^+(x)$は節点xより出る枝集合、$\omega^-(x)$は節点xへ入る枝集合を示す。$N=[G, I, f]$で表わす。

許容フロー：各枝を流れるフローが、枝の閉区間に入っているフロー。

フローコスト：単位当りの枝流量を維持するのに必要なコスト率が、各枝に定数重み$r(e)$として与えられているときのフローfの総コスト

$$\left(\sum_{e\in E} r(e)\cdot|f(e)|\right)$$

隣接する：有向グラフGに於いて、$\Phi(e_m)=(v_i, v_j)$ $v_i, v_j\in V$ なるe_mが存在するならば、v_iはv_jに隣接しているという。

到達する≦：Gに於いて、全ての$i=1\ 2\ 3 \cdots m-1$に対して，v_iがv_{i+1}に隣接している頂点の連続$v_1 v_2 v_3 \cdots v_m$が存在するならば、v_1はv_mに到達可能という。

パスJ(順枝列)：各節が次の節に隣接している節の連続(各付向枝の始節が次の付向枝の終節になるような付向枝の連続。並列枝の無いネットワークでは、節の連続で表わしてさしつかえない)

有向サイクルL(順閉路)：始めと終りの節点が同一の順道。

単位順枝列フローf_J：順枝列に含まれる付向枝では$f(e)=1$，それ以外では$f(e)=0$

フローマトリックスF：

$$f_{ij}=\begin{cases} f(e_m) & v_i\text{が}v_j\text{に隣接しており、}\Phi(e_m)=(v_i\ v_j) \\ 0 & \text{それ以外}\end{cases}$$

強連結成分：どの$v_i, v_j\in V$に対しても$v_i\leq v_j$，$v_j\leq v_i$ かつ、Gの極大な部分グラフ

3. 最少で最短なS−Tパス集合の導出[3]。

この節は、有向グラフGの全ての有向辺を含む最少で最短なS−Tパスを導出する方法を述べる。

この方法は、グラフの各辺にフローの下限として1，コスト率として1を与え、そしてこのグラフの最少コストフローを構成し、そしてそれから、フローマトリックスを単位順道フローと単位順閉路フローに分解、結合して、最少で最短なS−Tパスを求める。

次に上記の方法で最少で最短なS−Tパスが求められることを述べる。

(1)、任意の最少フローは、S−T単位順枝列フローに分解出来る。

(2)、最少フローのS−T単位順枝列フロー分解に対応するS−Tパス集合は、Gの全ての有向辺を含む最少のS−Tパス集合である。

(3)、最少コストフローのS−T単位順枝列フロー分解に対応するS−Tパス集合はGの全ての有向辺を含む最少で最短のS−Tパス集合である。

3.1、最少コストフローの構成

1、許容フローを構成する。

2、SよりTへ最少フローを流す。

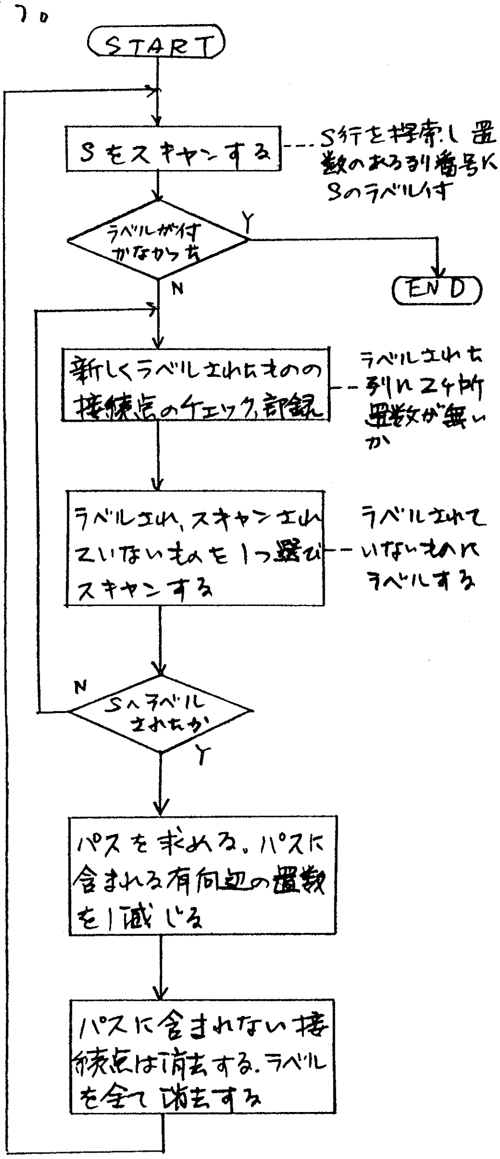
3、最少コストフローをフローマトリックスに構成する。

3.2、フローマトリックスよりパス集合の導出

フローマトリックスから、最少で最短なS−Tパス集合の導出は次のように行う。

上記により導出された最少コストフローマトリックスに図，1の手続きを

行う。



図．1

図．1の手続の終了時点で、未だ出入フローのある全ての接続点をSとし行と列をかえて、図．1の手続を同様に行う。

上記の手続が終了した時点で未だ出入フローのある接続点があったら、さらに同様に上記手続を行う。

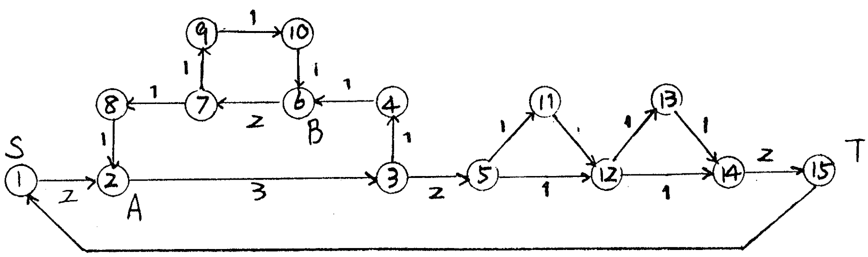
これらの手続を繰返し、接続点が全く生じなかった段階で終了する。

図．2は最少コストフローが構成されたネットワークの例を表わしている。付向枝に付けられた値は、フロー値を表わしている。S－Tパス集合の導出は、第1段階で例えば、(1 2 3 5 11 12 14 15)、(1 2 3 5 12 13 14 15)のパスが導出される。接続点は2つの有向枝が入っている節のため、この場合はA点となる。第2段階で、(2 8 7 6 4 3)が導出され、逆向に変換されて、上記のパスのいずれかに連結される。例えば、(1 2 3 4 6 7 8 2 3 5 11 12 14 15)というパスが構成される。接続点はBとなる。第3段階で、(1 2 3 4 6 7 9 10 6 7 8 2 3 5 11 12 14 15)(1 2 3 5 12 13 14 15)の最少のS－Tパスが得られる。

4. 未テスト辺を含む最少で最短なS－Tパス集合の導出。

4.1. 有向サイクルを含まないプログラムグラフの場合。

有向サイクルが含まれないグラフの最短な未テスト辺を含む最少パス集合の導出法について述べる。



図．2

この方法は、図.3のようにして導出する。

未テスト辺にフローの下限1を与え それ以外の辺に0を与え、許容フローを導出する

↓

各辺にコスト1を重みとして与え S−T最少フロー時の最少コストフローを導出する。

↓

フロー分解の手続を行う

図.3

図.4の例題でこの導出法を述べると、

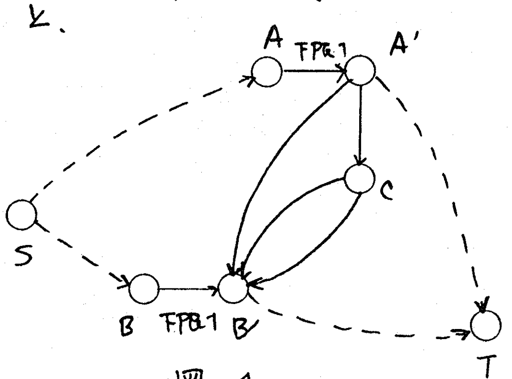


図.4

例えば、許容フローは点線のように求まり、最少フローはTからSへ流せるだけ流すことにより求まり、最少コストフローはコスト率が負の閉路へのフロー重畳により求められる。

4.2、有向サイクルを含むのを許したプログラムグラフの場合

有向サイクルを許したグラフの最短な未テスト辺を含む最少パス集合の導出法について述べる。

先に示した最少コストフローによる方法を直接適用すると、図.5のように、例えば許容フローは図となり、最少フロー導出（2本のTからSへのフローを流す）によりフローは0となることが生じる場合があり、目的のパスを導出することは出来ない。

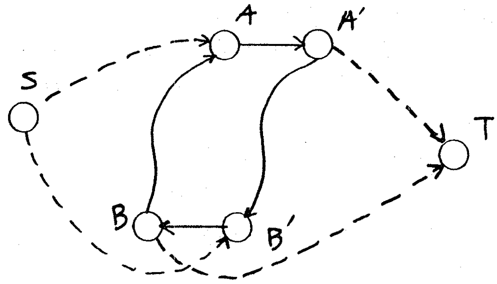


図.5

このため、図.6のように、

強連結成分を検出する

↓

未テスト辺を含む各々の強連結成分について次のことを行う

1) 強連結成分の入口と出口並に、その強連結成分内に含まれる未テスト辺を節とし、その到達関係を辺とするグラフを構成し、辺に最短距離を重み付ける
2) 入口節から出口節へのハミルトンパスの最少のものを求める。
3) ハミルトンパスの節間の元のグラフ上での最短パスを求める。

↓

未テスト辺を含む強連結成分を前のステップで導出したパスに置きかえたプログラムグラフにおいて、未テスト辺に下限1、それ以外に下限0 全ての辺にコスト率1を与え、S−T最少フロー時の最少コストフローを導出する

↓

フロー分解の手続を行う

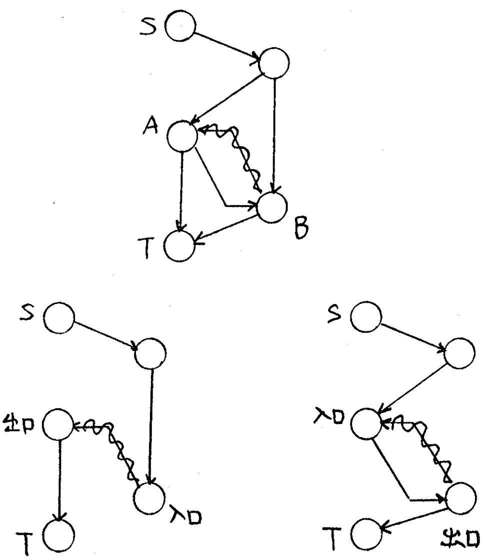
図.6

ハミルトンパスの最短のものを導出するアルゴリズムを応用して解くことが

考えられる。

ところでこの方法は、ハミルトンパスの最短なものを求めるアルゴリズムを含んでおり、強連結成分内の未テスト辺の数が多いとき（例えば、10以上）は余り効率の良い方法は見い出されていない。

又、プログラムが構造化されていないとき、入口と出口の組合せの対により、未テスト辺を含む最短なパスの長さが異るため（図．7）、



図．7

入口と出口の組合せに対して、※の手続を行う必要がある。

しかし、この方法は未テスト辺は本来数が少ないことを考慮すると、実行上は有用な方法と思われる。

ところで、もし、強連結成分内に未テスト辺が直列に連続的に生じている場合は、この方法の※の部分を図．8に変えることにより、ハミルトンパスを求めずに簡便に求める方法が考えられる。

強連結成分内の未テスト辺の連続性のチェックは、強連結成分の検出の際に未テスト辺を記憶し、グラフの隣接行列を用いて調べることにより、チェックが可能となる。このため、このテスト法の適用は、いくつかのテストの過程で、上記の条件が満足されたところで適用すればよいことになる。

未テスト辺を含む各々の強連結成分について、入口から未テスト辺の連続の尾の辺へ、未テスト辺の連続の頭の辺から出口への最短路を求める

図．8

ハミルトンパスの最短なものを求めるのは、一般には難解であることと、テストパスは必らずしも最短であることは要求されないことを考慮して、次に有向サイクルを含むのを許したグラフの未テスト辺を含む最少なパス集合を効率的に求める方法を述べる。この方法のA案とB案は、次のようになる。

強連結成分を検出する

↓

未テスト辺を含む各々の強連結成分について、入口から未テスト辺への最短パス、未テスト辺から出口への最短パスを求める。※※

↓

プログラムグラフの未テスト辺並に前ステップで導出されたパス上の辺に、フロー下限1，その他に0、全ての辺にコスト率1を与へ、S－T最少フロー時の最少コストフローを導出する。

↓

フロー分解の手続を行う

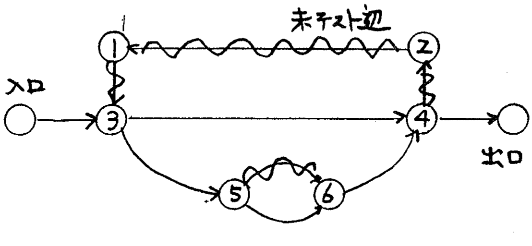
A案

図．10

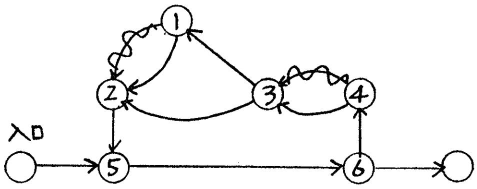
上記A案の、＊＊の部分を次に変えるとB案となる。

未テスト辺を含む各々の強連結成分について、入口から未テスト辺を順々に近いものから近いものへと進み出口まで辿るパスを求める

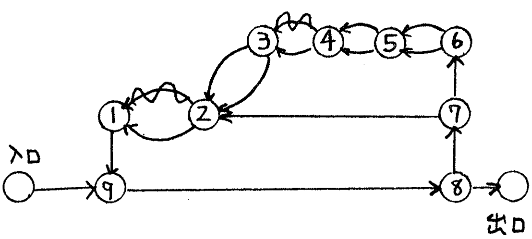
A案とB案の最短路を得る可能性の大・小については、未だ検討しなければならないが、A案の導出法はB案より、ラベリング法を用いて効率よく行うことが出来る。



A，B案　可



B案　可



A，B案共不可

図．11

5. 参考文献

[1]. R.K. Deb, On Generation of Test Data and Minimal Cover of Directed Graph, IFIP Congress Proceedings, pp13-16, 1977.

[2]. S.C. Ntafos and S. Louis Hakimi, On Path Cover Problems in Digraphs and Applications to Program Testing IEEE Transaction On Software Engineering, Vol. SE-5 No5, pp520-529, 1979.

[3]. 柳沢隆夫，プログラムテストに用いるパスジェネレータへのグラフ理論の応用（第1報），芝浦工大研究報告理工系編，第30巻，第1号，1986.

[4] R.E. Prather, Theory of Program Testing－An Overview, THE BELL SYSTEM TECHNICAL JOURNAL, Vol. 62, No.10, part2, pp.3073-3105, 1983.

[5]. 柳沢隆夫，パスカバー法によるプログラムテストデータの自動生成法に関して，芝浦工大研究報告理工系編，30巻，第2号，1986.