

プログラムテストに用いられる——
パスジェネレータへのグラフ理論の応用研究
Application of Graph Theory Algorithms to——
Path Generator for Program Testing

柳沢隆夫
Takao Yanagisawa
芝浦工業大学
Shibaura Institute of Technology

本論文は、プログラムの自動的なテストパス作成に就いて生じる、3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と、有向グラフの指定された節とパスを通る最少のパス集合を
決定することである。

In this paper we consider algorithm for three problems
that arise in automatic test path generation for pro-
gram: 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 path which
is through a specified edges of a directed graph and
the problem of determining a minimal set of path whi-
ch is through a specified node and path of directed
graph.

[1]. はじめに

プログラムテストは、プログラムの信頼性を
高めるために行う。プログラムの全ての、スタ
ートからエンドまでの経路を求めて、この経路
を通るテストデータを算出して行う完全なテ
スト方法は、普通のプログラムでも、そのような経
路の数は膨大となり、そのテストが実行不可能
となることがあるため、全ての経路集合の部分
集合を求めてテストを行う方法が考えられてい
る。この部分集合の選定方式、ある基準に合致
するものを選ぶものがあり、基準にも色々な
ものがある。(図1)

本研究は、以下に示す3つのテスト選択基準

を満足する、テスト経路の導出法について検討
する。(ここで、プログラムをプログラムクラ
フに変換(1)して、導出法を求める)

(1)プログラムの各々の分枝を少なくとも1度は
実行するテスト経路集合の選定方式⁽²⁾⁽³⁾

本研究は、有向サイクルが含まれることを
許したプログラムグラフの、各有向辺を含む
最少のパス集合を求め、さらに有用性を増す
ために、上記の内でも最短(パス集合に含まれ
る有向辺の延数が最少のもの)に属するパス集
合を求める解法について述べる。

(2)各々の未テスト命令を少なくとも1度は実行
するテスト経路集合を選ぶ

プログラムのテスト経路を先ず決定して、

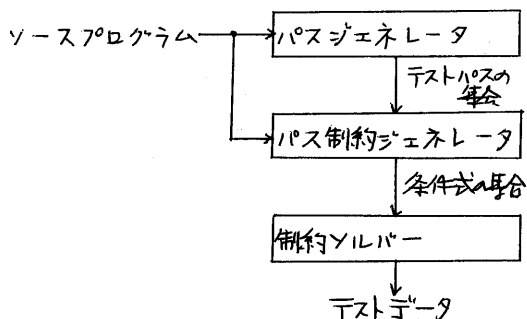


図. 1

次に、これを通る入力データを算出して行うテストは、そのテスト経路が実行経路となり、破局となる。この危険性を柔らげるものとして、先ず、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、未だ未テストの部分が有ったとき、そこにテストを振り回すテスト法が考えられる。

そこで、本研究は、有向サイクルが含まれることを許したプログラムグラフの、各々の未テスト辺を含み、最短な最少数パス集合が、プログラムが構造化されている場合について効率的な方式が見い出されたので、その解法について述べる。

プログラムが構造化されていない場合は、プログラムを構造化されたものに変換するプログラムを使用することにより、上記の方法を適用すれば、解法は可能となる。

(3). 各々のデータフロー・アノマリーを少なくとも一度は実行するテスト経路集合の選定⁽¹⁾

プログラムに含まれる、ある変数が、定義された後、そのプログラムαの経路に沿って、その次の使用のまゆネが定義であるようになると、データフロー・アノマリーが生じるといふ。このため、データフロー・アノマリーを含むテスト経路は、誤りが含まれていることが多いことが予想される。

そこで、本研究は、有向サイクルを含むプログラムグラフの、アノマリーに対応する節、並にパスを含む、最短な最少数パス集合を求める解法について述べる。

[2]. 最短な最少数パス集合の導出解法

2-1. 各々の有向辺を含む最短な最少数パス集合の導出解法

手順

step. 0 グラフの各々の辺にフローの下限
1. コスト率を1と定める。

step. 1 S→Tフロー最少数時の最少数コストフローを構築する

(プログラムグラフであることから、スタートノード、Sのみが入次数0、エンドノード、Tのみが出次数0、であるため、各辺がSより到達出来、又Tへ到達出来るため、SからTへ、フロー、1を流すことにより、許容フローは得られる。

Step. 2 S→T単位増道フローと、単位増閉路フロー分解、結合して、最短な最少数パス集合を導出する。

(グラフ理論で、グラフの全てのフローが正のときは、正の値閉路フローに分解出来ることは証明されている。プログラムグラフは単結グラフのため、これらの正の値閉路は、SからTへの増道フローと、直接かあるいは他の閉路を介して、接続点を有している。このため、この接続点を記憶しながらSとTと、連結を進めていくと、グラフの正のフローは、単位増道フローに分解出来ることとなる。(図. 2, 図. 3)

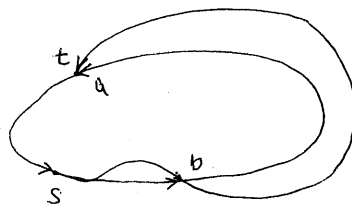
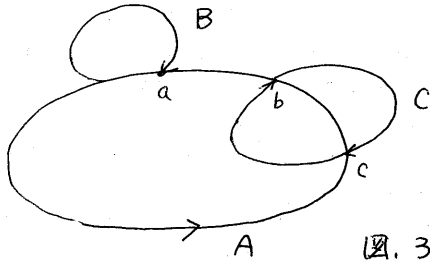


図. 2

TからSへ有向辺を設け反上で、Sよりラベルリング法を用いてパスを拡張していくと、Sへ戻る閉路フロー分解は、見い出される。これがSからTへの増道となる。パスを求める過程で、入って来る有向辺を接続点として記憶するが、上記の閉路が見い出された時点で、閉路フローをグラフより抜き取ると、a, bはフローのない

接続点となって消滅する。

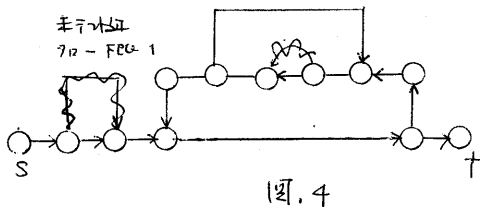


閉路間の連結は、接続点を用いて行われるが、閉路Aのフローが抜き取られる後、a, b, cがその接続点となる。例えば、bを接続点として選んだ場合、cを抜き取るとcはフローが無くなり、消滅する。

プログラムのS-Tパスに沿ってフローを流すと、上記の各有向辺を含むパス集合に対応するフローは、許容フローとなる。このため、SからTへの最少フロー-許容フローに対応するパス集合は、各有向辺を含む最少パス集合となる。最少フローが最少コストフローであったとき、S-Tフローの重複部分が最少であることを意味するため、最少コストフローのフロー分解は最短な最少パス集合となる。

2-2. 未テスト辺を含む最短な最少パス集合の導出解法

プログラムグラフに、有向サイクルが含まれていないときは、未テスト辺にフローの下限1全ての辺にコスト1を割り充て、2-1で用いた方法を応用することにより、導出される有向サイクルがプログラムグラフに含まれていると、最少コストフローの導出の段階で、分離した閉路フローが生じ、(図. 4)この解法は適用出来ない。



(1). ハミルトンパス導出を含めた解法(近似解) プログラムが構造化されているときは、次の方法が考えられる。

手順.

step. 1

D, F, S法を用いて、Sより有向辺の探索を行い、リターン辺の削除をする。(これにより、グラフより有向サイクルは、消滅させている)

強連結成分の入口(あるいは出口)にフローの下限1を設ける

step. 2

未テスト辺に下限1, 全ての辺にコスト1を割当て、S-Tフロー-最少時の最少コストフローを導出、分解・結合をへて最小パス集合を求める。

step. 3

step. 2で未テスト辺とされた辺は全て、テスト済みとした上で、グラフの各々の未テスト辺を含む強連結成分を検出し、その各々の成分の入口から、出口までの未テスト辺を含む最短なパスを導出する(各々の成分に、最短なハミルトンパスを求める解法を応用する)

step. 4

step 2で求めたパスに、step 3で求めたサイクル内のサブパスを連結させる。

(2) 分離フローを修正することによる解法 プログラムが構造化されているとき、S-Tフローから分離した閉路フローのハターンには、次のものがある。

(a). 未テスト辺を含むレベルパスと、それよりレベルが下び、その2つの端点を含むレベルパスの1部が生じる。

(b). 未テスト辺を含むレベルパスと、それを股いでいる、ひとつ上のレベルパスの1部の間が生じる。

このため、(a)のフローが生じないために、図5のように、ひとつ下のレベルの入口の辺(あるいは出口の辺)を未テスト辺の扱いにして、フローの下限1を与える。(b)のフローが生じないために、図. 6のように、未テスト辺を1番内側で股いられレベルパスの入口の辺(あるいは出口の辺)を、未テスト辺の扱いにして、フローの下限1を与えることが考えら

れる。

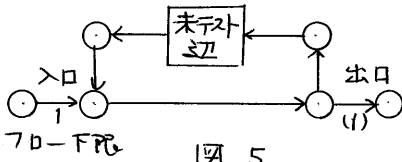


図. 5

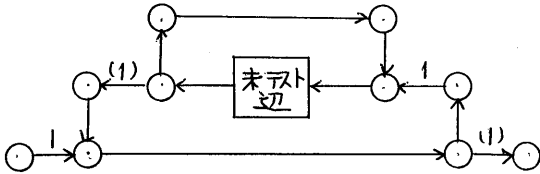


図. 6

未テスト辺を含む、最短な最少パス集合の導出手順は、次のようになる。
手順

step. 0 未テスト辺にフローの下限1, 全ての辺にそれぞれコスト1を与える。

step. 1 D, F, S 法で、各々の強連結成分を検出、未テスト辺の検出を行う。未テスト辺を含む成分については、入口(あるいは出口)に、フロー下限1を設ける。

step. 2 未テスト辺を含む各々の強連結成分について次を行う。

2-1 レベルハスを導出して、最高次レベルに含まれる未テスト辺を検出する。

(a) 未テスト辺を1番内側に取った形の1つ上のレベルハスの間に有る閉路を見出し、最少コストフローを導出する。このレベルハスの入口の辺(あるいは出口の辺)にフロー下限1を設ける

(*許容フローは、未テスト辺の方向に沿ったパスを拡張した節点と、逆方向に拡張した節点と一致したところで閉路を見出して導出出来る。コスト改善の前に、その閉路の最少フローを導出すると効率が良い)

(b) 未テスト辺を含むレベルハスの入口の辺(あるいは出口の辺)にフロー下限1を設ける
(*を行う)

2-2、step 2-1 の操作により、未テスト辺とされた現在の最下次のレベルハスが0のとき終了。そうではないときは、最下次のレベルハスを step 2-1 の最高次レベルハスとして、Go To step 2-1

step 3 グラフ全体のレベルハスの閉路を見出し、最少コストフローを導出する

step 4 step 3 までに見い出された各閉路について、さらにコスト改善して、最少コストフローを得る。

step 5 フロー分解 結合の手順により、最短な最少パス集合を得る。

2-3 アノマリーの節並にパスを含む、最短な最少パス集合の導出解法

アノマリーのカットパスは、どれか1つを含めばよいとすると、次のようなアルゴリズムが考えられる。

手順

step 0 アノマリーの節は、有向辺に置換え、フロー下限1を与える。アノマリーの節に入って来る有向辺は、アノマリーの有向辺の始節へ連結、出る有向辺は、アノマリーの終節へ連結させる。(図. 7)

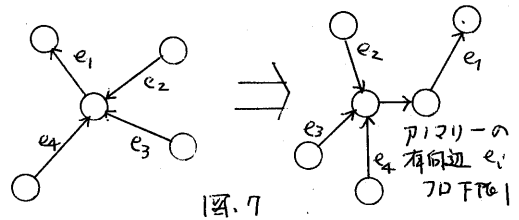
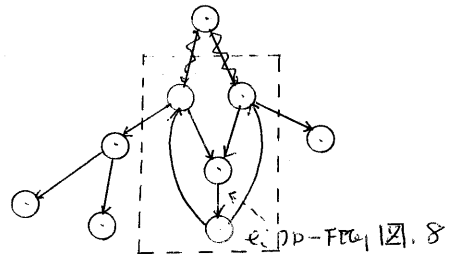


図. 7

アノマリーのパスは、新たに有向辺 e_i を設け、フロー下限1とする。各々のアノマリーのカットパスの終節から e_i の始節へ有向辺を設ける。 e_i の終節から各々のカットパスの終節へ有向辺を設ける。(図. 8)



アノマリーの有向辺 e_i , サブパスに含まれる辺にフローの下限 l_i を与え、アノマリーが重複しているときは、その傾数をフローの下限とする。

構造的なヒューマンエラーの全この辺にコスト 1 を与える

Step. 1 最長フロー時の最長コストフローを導出する。

Step. 2 フロー分解、結合の手続きを行う。
(異なるアノマリーがいくつもある辺に重複しているとき、辺に各々のアノマリー種類別の印を付けて、S-T フローの導出方法を決定する必要がある (図. 9))

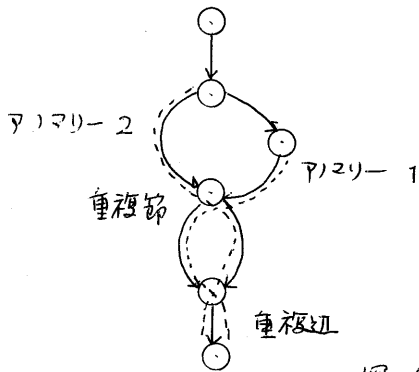


図. 9

[9] 参考文献

- (1) 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 (オノ報), 茨浦工業大学研究報告理工系編, 30巻, 31号, 1986.
- (2) 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, 茨浦工業大学研究報告理工系編, 30巻, 32号, 1986.
- (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の作成法, 情報処理学会研究報告, ソフトウェア工学, 63-4, 1988.