

プログラムテストに用いるパスジェネレータの作成法

柳沢隆夫
芝浦工業大学

この論文において、有向グラフの指定された辺を含む、最少のパス集合を決定する問題を検討する。この問題は、プログラムの各々の命令を少なくとも一度実行する、コンピュータプログラムのテストデータを発生させるのと同値となる。

Derivation of Path Generator for Program Testing

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

In this paper we consider the problem of determining a minimal set of path which contain the specified edges of a directed graph. This problem is equivalent to generating test data for a computer program, which executes each instruction in the program at least once.

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テスト法(図. 1)は、普通のプロ

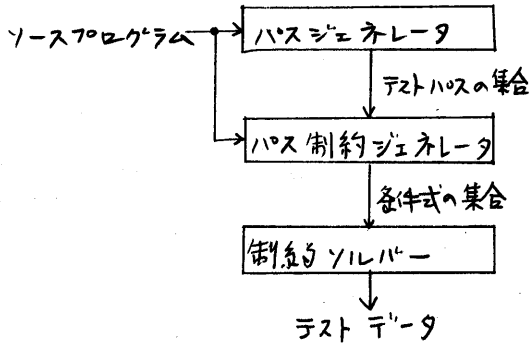


図. 1.

グラムでも、そのような経路の数は膨大となり、完全なテストは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。この部分集合の選出に、ある基準に合致するものを選ぶものがあり、基準にも色々あるものがある。この基準として、プログラムの各々のブランチを少なくとも一度は実行するテスト経路集合を選ぶものがある。

このテスト経路の導出法についてはいくつかの研究が報告されている。しかし、この研究はプログラムに有向サイクルが含まれていない場合に限定して行っている。

1例、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に対する最少パス集合の導出について、これまで若干行われている。Prather は、正則式を用いて導出した。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場合適用出来ない。

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(構造化されていないものを含む)の各有向辺を含む、最少パス集合を、グラフ理論を応用して求める問題を扱い、又有用性を増すために、上記のパス集合の重複部分を最少にしたパス集合(最短のパス集合)を求める解法について述べる。

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導出して行うものは、テストパスが不実行パスであったときは破局となる。

この危険性を減らすものとして、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、未テストの部分にテストを振り回すテスト法が考えられている。

しかし、このテスト法を具体的に定めた研究は余り見受けられない。ここでのテストは、最少回のテストで全ての未テスト辺を実行するテスト経路を導出して、効果的にテストすることが考えられる。

そこで、本研究は、未テスト辺を含めて、かつ最少のS-Tパス集合を求める問題を扱い、又実用上、有用と思われるので、上記の内で、既にテストされている有向辺が最少になる(最短なパス集合)テストパスの集合を、プログラムグラフが構造化されているときと、そうでないときに分けて、その導出法を検討する。

2. 各々の有向辺を含む、最少で、最短なパス集合の導出法

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、最少で最短なS-Tパス(スタート節から、エンド節までのパス)集合の導出法について述べる。

グラフの各々の辺に、フローの下限

1. 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構築し、次に、このフローを単位順道フローと単位順閉路フローに分解し、そして結合することにより、最少で、最短な $S-T$ パス集合が導出される。

分解、結合の手続は、手順. 1 に示される。

手順. 1.

step. 1 S をスキャンする (S 行を探作し、置数のある列番号に S のラベル付す)。もし、ラベルが付かなかつたならば終了。

step. 2 新しくラベルされたものの接続点の4エック、記憶 (ラベルされた列に、2ヶ所置数がないか4エック)

step. 3 ラベルされ、スキャンされていないものを選び、スキャンする。もし、 S のラベルが付かなかつたならば、step. 2 へ行く。

step. 4 パスを求める。パスに含まれる有向辺の置数を1減じる。パスに含まれない接続点並に全2のラベルは消去する。step. 1 へ行く。

この方法で求められる理由は、

(1)、プログラムグラフをレベルパスに分解すると、各レベルパスは、その1つ下のレベルの、同一のレベルパスの上に両端点を持っている。

(2)、 S からラベル化法を用いるならば、 S を含む順閉路が存在するとき、その閉路の1つを選んで導出する。

のためである。

3. 未テスト辺を含む、最少で最短なパスの導出法。

3.1 ハミルトンパスを導出するに与える解法。

この $S-T$ パスは、プログラムグラフの強連結成分を、その成分内の未テ

ストの有向辺を全て含み、かつ最短なパス (最短なハミルトンパスを求めろ問題に帰せしめて解く (手順. 2)) に置き換えたグラフにおいて、前記と同様に、最少コストフローを導出、フロー分解、結合して、解かれるものである。

手順. 2

step. 1 強連結成分を抽出する。

step. 2 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。

step (1)、強連結成分の入口と出口並に、その強連結成分に含まれる未テスト辺を節とし、その到達関係と辺とするグラフを構成し、辺の最短距離を庫み付ける。

step (2)、入口節から出口節への最短なハミルトンパスを求める。

step (3)、ハミルトンパスの節間のおとのグラフ上での最短経路を求める。

この方法は、プログラムが構造化されているときは、上記のハミルトンパスにより求められたパスと、強連結成分を入口から出口へ最短経路で通りぬける (未テスト辺に下限; 1 を設けない) パスを並列に設けることにより実現される。(図. 2) プログラムが構造化されていないときは、複数の入口から複数の出口へのフローで、未テスト辺がカバーされることなどあり、この方法は必ずしも成功しない。

ハミルトンパスを求める問題の解法は、いくつか提案されているが、グラフの節の数が多の場合、計算量が多大に増加する。強連結成分内の未テスト辺が連続して生じている場合 (プログラムが構造化されていないときは適用出来ない) のみ、強連結成分の入口から、未テスト辺の連続の尾の辺への最短経路と、連続の頭から強連結成分の出口への最短経路を求める簡単な方法

で、未テスト辺を含む最短経路が求められる。

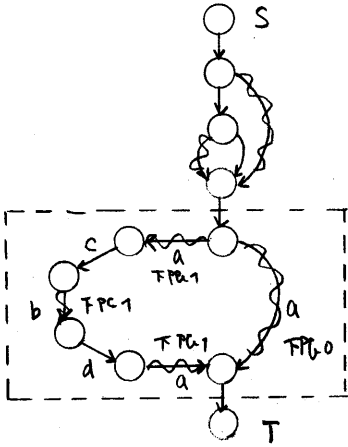
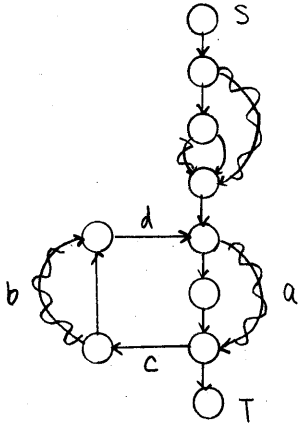


図. 2

3.2 最小コストフローを導出することによる方法

有向サイクルが含まれない、プログラムグラフの場合の導出法は、手順. 3で示される。

手順. 3

step. 0 未テスト辺にフローの下限 1 以外以外の辺にコストを与えない。

各々の辺に、コスト率 1 を与える。

step. 1 S-T 最小フロー時の最小コストフローを導出する。

step. 2 手順. 1 を用いて、フロー

分解、結合して、未テスト辺を含む、最少で、最短経路集合を導出する。

しかし、この方法は、プログラムグラフに有向サイクルが含まれているときに適用すると、図. 3 (S から T への最小フローを導出したとき、フローは実線で示されたところのみとなり、導出不能となる) となる場合が生じよう。

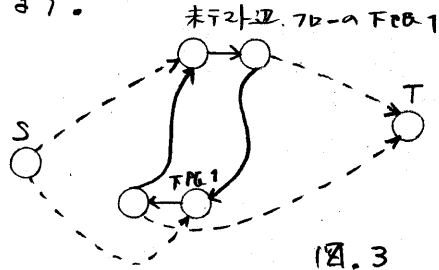


図. 3

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順閉路フローのパターンに次のものがある。

1). 未テスト辺を含むレベルフローと、それより 1 つレベルが下で、その 2 つの端点を含む、レベルフローの 1 部の間で生じる。

2). 未テスト辺を含むレベルフローと、それを覆っている、ひとつ上のレベルフローの 1 部の間で生じる。

このため、1) のフローが生じないために、図. 4 のように、ひとつ下のレベルへ入口の辺 (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。

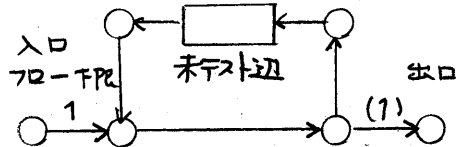


図. 4

2)のフローが生じないために、図. 5
のように、未テスト辺を含むレベルパス
の始めの辺(あるいは終りの辺)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。

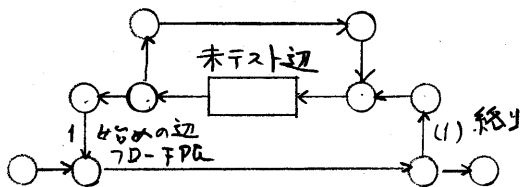


図. 5

このため、プログラムが構造化され
ているときの、未テスト辺を含む、最
少で、最短パス集合の導出法は、次
のように示される。

手順. 4

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

step. 1 最少コストフローを導出する。

step. 2 0. F. S 法を用いて、フロー
の有る辺を操作 (S より開始する)
し、分岐フローを検出する。もし、
検出されなかったときは、GO TO
手順. 1。

step. 3 レベルパスを導出して、最高
次レベルに含まれる未テスト辺を検
出する。(レベル 0 より、順にレベル
を上げ、未テスト辺が全て検出され
たところで終了)。次の事項を行う。

- 1), 未テスト辺を含むレベルパスの
入口の辺(あるいは、出口の辺)
に、フローの下限 1 を設け、以後
未テスト辺の扱いをする。
- 2), 未テスト辺を股いた形の 1 つ上
のレベルパスがあるときは、この
未テスト辺を含むレベルパスの始
めの辺(あるいは、終りの辺)に
フローの下限 1 を設け、以後
未テスト辺の扱いをする。

step. 4 step. 3 までの操作により、
未テスト辺とされた、現在の最下次
のレベルパスがレベル 0 のとき、
GO TO 手順. 1。そうでないとき、
最下次のレベルパスを step. 3
の最高次レベルパスとして、GO TO
step. 3

70プログラムが構造化されていないま
のを許す方法として、手順. 4 を適用す
ると、次のような場合、最短なテスト
パスは導出されない。

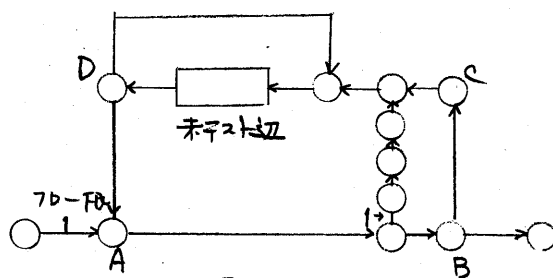


図. 6

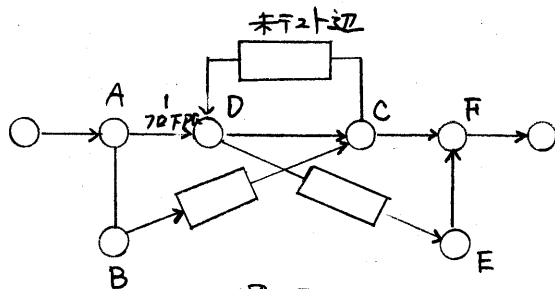


図. 7

図. 6 において、A B C D の最適パス
が導出されない。

図. 7 において、A B C D E F の最適
パスが導出されない。

4 参考文献

- [1]. R. K. Deb, On Generation of Test Data and Minimal Cover of Directed Graph, IFIP Congress Proceeding, pp 13-16, 1977.
- [2]. S. C. Ntafos and S. Louis Hakimi, On Path Cover Problems in Digraph and Applications to Program Testing, IEEE Transaction On Software Engineering, Vol. SE-5 No 5, pp 520-529, 1979.
- [3] 柳沢隆夫, プログラムテストに用いるハスジェネレータへのグラフ理論の応用(オ1報), 芝浦工業大学研究報告理工系編, オ30巻, オ1号, 1986.
- [4] 柳沢隆夫, ハスカバ-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, 芝浦工業大学研究報告理工系編, 30巻, オ2号, 1986.
- [5] 柳沢隆夫, プログラムテストに用いるハスジェネレータ作成に関する研究, 情報処理学会研究報告, ソフトウェア基礎論 20-3, 1987.
- [6] 柳沢隆夫, プログラムテストに用いるハスジェネレータへの考察, 情報処理学会研究報告, ソフトウェア工学 55-1, 1987.
- [7]. R. E. Prather, Theory of Program Testing - An Overview, THE BELL SYSTEM TECHNICAL JOURNAL, Vol. 62, No. 10, part 2, pp. 3073-3105, 1983.