アルゴリズム 6－1
（1989. 3. 22）

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考察

柳沢隆夫
芝浦工業大学

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、有向グラフの指定された辺を最も多く含むパスを決定することと、指定された個数で、有向グラフの最も多くの辺を含むパスの集合を決定することである。

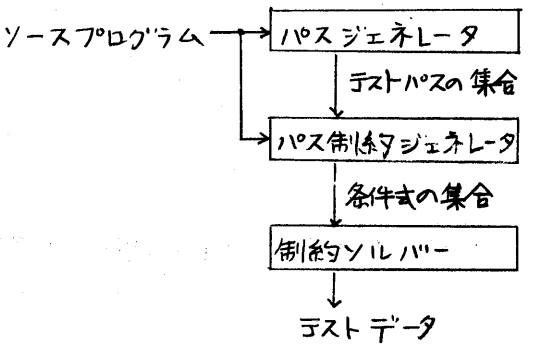
Application of Graph Theory Algorithms to Path Generater for Program Testing

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

In this paper we consider algorithms for two problems that arise in automatic test path generation for program: the problem of determining a path which is through the most specified edges of a directed graph and the problem of determining a set of path which is contain the most edges of a directed graph by specified times.

[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、その下スト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。この部分集合の選び方に、ある基準に合致するものを選ぶものがあり、基準にも色々なものがある。図－1



図－1

本研究は、以下に示す2つのテスト基準を満足する、テスト経路の導出法について、プログラムをプログラムグラフに変換した上で、検討をする。

(1)、プログラムの未テスト命令を最多に含むテスト経路。

プログラムの下スト経路を先ず決定して、次に、これを通る入力データを算出して行うテストは、その下スト経路が不実行経路のとき、破綻となる。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、未だ未テスト部分があったとき、そこに下スターを振り向けるテスト法が考えられる。[1]

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の、未テスト辺を最多に含み、かつ、その中でも最短となるS－Tパスの導出問題を扱う。(2)(3)(4)

強連結成分内の未テスト辺を含む最短なパスは、最短なハミルトンパスを求めることにより得られるが、これを求める種々の導出法は、効率が余り良くないことが知られている。

本研究は 強連結成分内に、連続的に未テスト辺が生じているとき、プログラムが構造化されているときにかぎり、成分の入口から連続の先の辺までの最短路と、連続の頭から出口までの最短路を求めることによって、簡単に成分内のパスが求められることを明示する。

又、本研究は、近似解法ではあるがプログラムグラフを到達可能グラフに変換した後(到達性を保存するため)D.F.S法をグラフのスタート節より適用しながら、グラフのリターン辺を削除する方法を用いることにより、有向サイクルを除去し、このサイクルの無いグラフに最長パス導出法を適用する方法によって、最短な最多パスが求められることを明示する。

(2)指定された回数で、最多にプログラムの命令を含むテスト経路集合

テストの達成目標が、100%以下で与えられているもの、あるいは、テストの回数が指定されているようなテスト法が考えられる。

100%以下で、何%と指定された下スト法は、テストの回数をひとつづつ上げて行き、その指定された%に達したところで、その回数による最適なテストを行うことにより、目標は達成される。

そこで、本研究は、テスト回数が指定されているときに於ける、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、最多な未テスト辺を含むテストパス集合（最多の条件を満足した上で、最短なもの）の導出解法について述べる。

[2] 未テスト辺を最多に含む、最短なS－Tパスの導出解法

2－1 強連結成分内の最短な最多パス

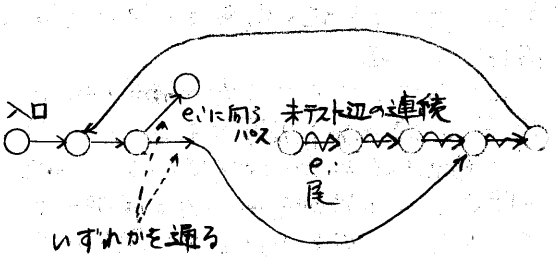
プログラムグラフに、有向サイクルが含まれていないときは、未テスト辺に大きな値（例えば、１００　）、既にテストずみの辺に－１を割り当、最長パスを導出することにより、最短な最多パスは導出されるが、サイクルが存在すると、パスはそれを何度も回ることになり、この方法は適用出来ない。

このため、強連結成分を別個に取り出して、未テスト辺を全て含み、かつ最短なサブパスを求めることが考えられる。

プログラムが構造化されており、強連結成分に未テスト辺が連続して存在しているとき、成分の入口から進み、未テスト辺の連続の途中に入るパスは連続の頭の辺を通るためには、連続に入り込んだときより、そのパスのさらに前の節に戻り、元のパスを進んで、連続に入り込んだ節を通るパスの先の節を通らなければならない。このため結局、入口から連続の尾までのサブパスに含まれる有向辺をこのパスは含むことになる。

又、連続の頭の辺から出口までのサブパスも、成分内の有向辺を逆向きにしてみると、同様のことが証明される。

このため、成分の入口から、未テスト辺の連続を通り出口へ進むパスは、成分の入口から連続の尾への最短路を通り、連続を通りぬけて、成分の出口まで最短路で通るパスが、成分内の最短なサブパスとなる。　図－２

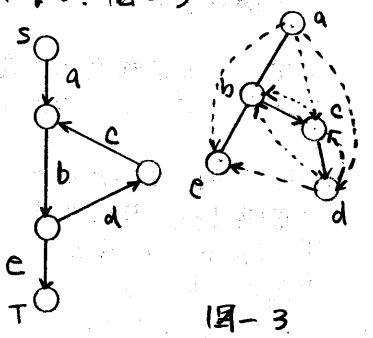


図－２

2－2 日.F.S法を用いた近似解法

日.F.S法を用いて、より効率的なアルゴリズムで、未テスト辺を最多に含むS－Tパスを求める方法について述べる。この方法はその内に、いくつかの修正アルゴリズムを加味することにより、より最短なパスに近ずけることが出来る。

有向サイクルを含んだ（サイクル内に未テスト辺が存在する）プログラムグラフの未テスト辺を含む最長パスはサイクルを含んでいる場合がある。そこで、未テスト辺を節とする到達可能グラフを作成して、最長パスを求めると、そこにはサイクルが含まれない、未テスト辺を最多に含むパスが全て存在することになる。図－３



図－３

図－３において、b,c,dをどの順に辿ろうと、必ず最後の辺からeへの辺が直接か、あるいはフォワード辺として存在する。

そこで、Sより任意に日.F.S法を適用して、リターン辺をグラフより消去すると、そのグラフにはサイクルを含

まない未テスト辺の最多なパスのみが存在している。

ところで、サイクル内の最多パスはB.F.S法の探索順序により、1つだけが選ばれると他のパスはリターン辺の除去により消滅する。例えば図-3で、サイクル内をb,c,dと選んだため他のb,d,cとc,b,dとc,d,bとd,b,cとd,c,bは全てb,c,dとは逆向きの辺(リターン辺)を含むことになる。このため、この方法は近似解しか得られない。

B.F.S法を用いた近似解法は、次のようになる。

手順.1

step.1 未テスト辺を節とする到達可能グラフの導出

step.2 到達可能グラフのSより、B.F.S法を用いて辺の探索をしリターン辺を除去する

step.3 SよりTへの辺の数の最多なパス(最長パス)を求める。

step.4 step.3のパスの欠落辺を求めて、未テスト辺の最多なパスを構築する。

手順.1を最短なパス導出に近ずけるために、次のような工夫が考えられる。

(1). step.2において、Sより各辺の最短距離を重み付け、B.F.Sの探索順序を辺の重みの小さい順の優先順位に従える。

(2) step.3において、各辺に(100-元のグラフの最短距離の重み付けをし、辺の重みの最長パスを求める。

(3) step.4において、最短路の欠落辺を求める。

[3] 未テスト辺を最多に含む複数パスの導出解法

3-1 未だテストされていないプログラムを対象とする解法

手順.2

step.1 強連結成分を見い出し、入口の辺にフローの上.下限:1を付け、最少コストフロー導出により、全ての有向辺を含む最短路を導出する

step.2 元のグラフに於いて、強連結成分をstep.1で求めたパスと、それに平行な、成分の入口から出口までの最短なパスに置き換える。

step.3 Sより出る有向辺に、指定されたパスの仰数のフローの上.下限を与える。

各辺にコストを次のように割当てる。

(1). 強連結成分に対して、

step.1で求めたパスの各辺にフロー:1のとき-100, フロー:2以上のとき0、成分の入口から出口までのパスの辺に-1

(2) その他の辺には

フロー:1のとき-100, フロー:2以上のとき-1

step.4 最少コストフローの導出、フロー分解、結合

手順.2はフローが2以上のとき、図-4に示すように、入口から出口までのパスがAを通るものと、通らないものが存在することにより、最短となるケースが生じてしまい、必らずしも最短にはならない。

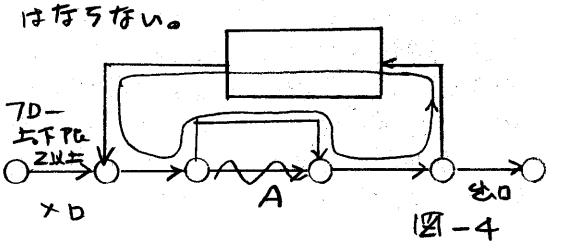


図-4

3－2 既にテストされ、未テスト部分が未だ残っているプログラムを対象とする解法

手順．2を次のように変えて導出する。

step.1は．未テスト辺を含む強連結成分について同様に行う

step.3の各辺のコストは次のように割当てる。

(1)．未テスト辺を含む強連結成分に対して

step.1で求めたパスの未テスト辺に、フロー:1のとき−100，フロー:2以上のとき0，その他の辺に−1、成分の入口から出口までの最短路の各辺に−1

(2)．未テスト辺を含まない強連結成分に対して

成分の入口から出口への最短路の各辺に−1

(3)．その他の辺に対して

未テスト辺にフロー:1のとき、−100，フロー：2以上−1，テスト済みの辺−1．

[4] 参考文献

(1). R.E Prather, Theory of Program Testing－An Overview, THE BELL SYSTEM TECHNICAL JOURNAL, Vol.62, No.10, partz, pp 3073－3105, 1983.

(2) 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一考察，情報処理学会研究報告，ソフトウェア工学，54－2，1987.

(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自動生成について，情報処理学会研究報告，アルゴリズム，3－6，1988.

(4) 柳沢隆夫，パスジェネレータの自動生成によるプログラムのテスト法に関して，情報処理学会研究報告，情報学基礎，11－4，1988.