『ループ領域』とその解析アルゴリズム

4S-4

牧野　正士
日本アイ・ビー・エム（株）　東京基礎研究所

1. はじめに

アセンブラやC言語で書かれたプログラム構造を理解するために重要な要因の一つは、そのループ構造を明らかにすることである。「自然な」ループの定義は、プログラムのコントロール・フローグラフ（以下フローグラフと略す）がreducibleなときには、interval partition 基づくもの[He77]や、[ASU87]に簡潔に、しかしやや informal に述べられているものなどがある。Ada や Pascal などのプログラムは全て reducible であり、また、Knuth は、実際ほとんどのFORTRAN プログラムは、reducible であると述べている[Kn71]。しかし、FORTRAN や C では、reducible でないプログラムが書けるし、アセンブラ・プログラムの解析においては、しばしば reducible でないフローグラフに遭遇する。また、Knuth 自身の著書の中[Kn73]にも、reducible でない「自然な」フローグラフの例を見出すことができる。しかし、reducible でないフロー・グラフのループ構造は、従来の定義や手法を適用することによっては調べることができない。本稿では、枝重み付き深さ優先探索図上のループ領域という概念を用いることにより従来の「自然な」ループの定義を一般の reducible でないフローグラフ上に拡張すると同時に、ループ領域を効率よく求めるアルゴリズム、及びそのインプリメント結果について述べる。

2. ループ領域

[定義1] フローグラフ G=(N,A,s)（N:ノードの集合、A:枝の集合、s:開始ノード）に対して、G が reducible であるとは次の1°、2° のいずれかの条件を満たすときをいう。

1° G は有向サイクルを含まない。

2° G の任意の有向サイクル C に対して、唯一の h ∈ C が存在し、s から C 内のノードに至る任意のパスは、h を通る、すなわち、C の任意のノードは、h によって dominate される。

図1、3の G_1, G_2 はいずれも reducible なフローグラフの例である。フローグラフの枝重み付き優先探索図（以後、'探索図'と略記する）とは、与えられたフローグラフ G の各分岐ノードに対して、そこから出る各々の枝に予め優先順位(重み)を与えておき、s を開始点とする深さ優先探索[AHU83]を、各分岐ノードで、優先順位の高い枝が優先的に選択されるように行ったときに 探索順序に応じて G の各枝を通常の tree arc, back arc, forward ark, tree arc に分類して表した図のことである 図2は、図1のフローグラフ

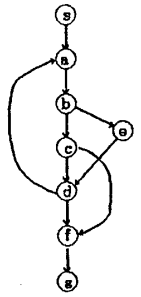

図1 G_1 =(N,A,s)

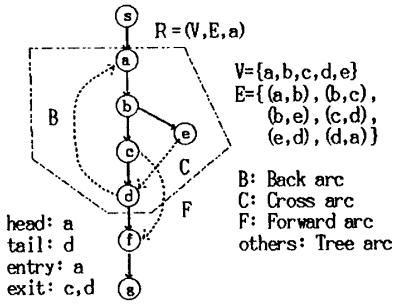

図2 G_1の深さ優先探索図とループ領域

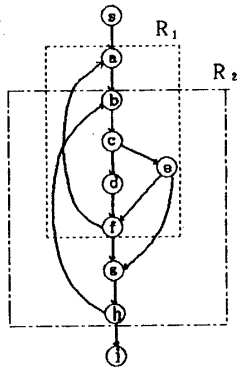

図3 G_2 =(N,A,s)

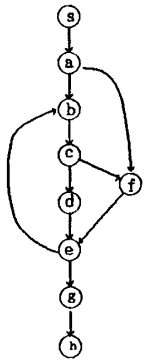

図4 G_3

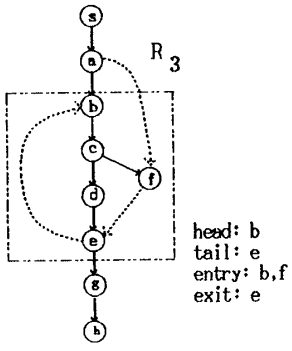

図5 アルファベット順を優先した時の G_3 の探索図

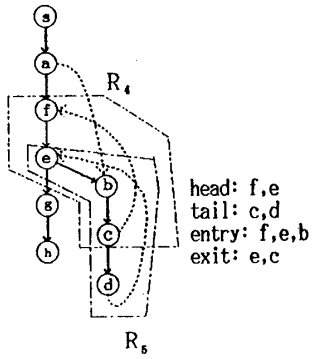

図6 アルファベットの逆順を優先した時の G_3の探索図

A generalized 'loop region' analysis
Seishi MAKINO
IBM Research, Tokyo Research Laboratory

の分岐ノードの各枝の重みを、分岐先のノードのアルファベット順に付けた時の探索図である。探索図において、各 back arc の始点を tail、終点を head と呼ぶ。

一般に探索図の形は、枝の重み付けによって変わるが、フローグラフが、reducible なときは、探索図の back arc の集合は、枝の重み付けによらず常に一定であることが知られている([HU74])。この性質により、reducible なフローグラフについての従来の'自然な'ループの定義

([ASU87],[He77])は、探索図上のループ領域として次のように述べなおすことができる。

[定義2] G の ループ領域とは、次の4条件を満たす R=(V,E,h) (ただし、V⊂N, E⊂A, h∈V)のことである
1° h は、G のある探索図の head である(h を R の head と呼ぶ)。
2° (t,h) ∈ E を満たす探索図上の任意の tail t に対して、t ∈ V (t を R の tail と呼ぶ)、かつ、back arc を経由しないで h から t に至る G のパス上の全てのノードは V の要素であり、これ以外に V の要素はない。
3° x, y ∈ V かつ (x,y) ∈ A ならば、(x,y) ∈ E 。

ループ領域を用いると、reducible なフローグラフのループをうまく特徴づけることができる。例えば、図1のフローグラフ G_1 のループ領域は、図2の探索図で、二点鎖線で囲った、ノード a, b, c, d, e から成る領域 R=(V,E,a)である。また、図3の G_2 の場合、ループ領域は、破線の領域 R_1と、一点鎖線の領域 R_2 の2つである

1° ~ 3° の定義から明らかに、ループ領域について次の性質が成り立つ。

(性質1) reducible なフローグラフのループ領域は、枝の重み付けとは無関係に、常に一定である。

フローグラフが reducible でない場合には、(性質1) は必ずしも成り立たない。例えば、図4のフローグラフ G_3 の探索図は、枝の重み付けによって変わる(図5,6)。そこで、我々は、一般のフローグラフのループ領域の定義を、探索図に依拠して、 定義2によって与えることにする。しかし、探索図の選び方に 無関係に次の性質が成り立つ

(性質2) フローグラフ G が reducible であるための必要十分条件は、G のある探索図上の任意のループ領域 R に対して、R 外のノードを始点とし、R 内のノードを終点とする、cross arc または forward arc (R への '飛びこみ枝'と呼ぶことにする)が存在しないことである。

フローグラフ G のループ領域 R=(V,E,h) において、R への飛び込み枝の終点と h を、R の entry ノードと定義する。また、G の探索図上で、R 内のノードを始点とし、R 外のノードを終点とする任意の枝を R からの ' 飛び出し枝 ' と呼び、その始点を R の exit ノードと定義する。定義2から明らかに、探索図上の各ヘッドと、ループ領域が一対一に対応する。そこで、R=(V,E,h)を R(h) と略記する。

3. アルゴリズムの概略

フローグラフ G の探索図上で、G の全てのループ領域および、その entry ノード、exit ノードを求めるアルゴリズムの概略を図7に示す。初期状態では、各 R(i) は空である。フローグラフ G=(N,A,s) およびその探索図が与えられているとき、ループ領域(s) を実行すると、各 i に対して、R(i) およびその head, tail, entry ノード、exit ノードが逐次計算される。

```
1. procedure ループ領域(v)
2. begin
3.    (v,w) が tree arc となるような
      v の各 successor w に対して、
4.    do begin
5.       現時点において、v ∈ R(i) を満たす、
         全ての R(i) に対して、w を R(i) の要素とする;
6.       ループ領域(w) をリカーシブに実行する;
7.    end;
8.    (v,w) が back arc となるような、
      v の各 successor w に対して、
9.    do begin
10.      w を R(w) の entry ノード とする。
11.      w から v までの tree arc をたどって得られる
         path 上の全てのノードを、R(w) の要素とする;
12.      v を R(w) の tail とする;
13.   end;
14.   この時点で v ∈ R(i) を満たす各 R(i) に対して、
15.   do begin
16.      v が successor を持たないか、または、
         v の全ての successor が R(i) の要素でないなら、
         v を R(i) の要素から削除する;
17.      v の successor の中に R(i) の要素と、そうでない
         ものが存在するなら、v を R(i) の exit ノードとする;
18.   end;
19.   (v,w) が cross arc または forward arc となるような、
      v の各 successor w に対して、
20.   do begin
21.      w ∈ R(i) かつ v∉R(i) を満たす全ての R(i) に対して、
22.      w を R(i) の entry ノードとする;
23.   end;
24. end;
```

図7 ループ領域解析アルゴリズム

4. アルゴリズムの計算量とインプリメンテーション結果

R(i) を表現するデータ構造および、8-22 行目の R(i) の参照と更新を無駄なく行えば、G=(N,A,s) のループ領域を、ランダム・アクセス計算機上で、O(k|N|+|A|log|A|) ビットのメモリと、O(k|A|) 時間で計算できる。ここで、k は、G のループ領域の数である。本アルゴリズムを、IBM 3081 システム 上で PASCAL/VS を用いてインプリメントした結果、実際の S/370 アセンブラ・プログラムから抽出された、ノード数1818, 枝数 2505, ループ領域数 50 の reducible でないフローグラフの全ループ領域及び entry, exit を約 0.6 秒で計算した。

謝辞 本研究の発端となった業務上のテーマ、並びに援助を下さった、日本アイ・ビー・エムの秋山義博氏に感謝します。

引用文献

[AHU83] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, 'Data structures and algorithms', Addison-Wesley 1983.
[ASU87] A.V.Aho, R. Sethi, and J.D. Ullman, 'Compilers', Addison-Wesley, 1987.
[He77] M.S. Hecht, 'Flow analysis of computer programs', North-Holland, 1977.
[HU74] M.S. Hecht and J.D. Ullman, 'Characterizations of reducible flow graphs', J.ACM, vol.21, no.3, 1974.
[Kn71] D.E. Knuth, 'An empirical study of FORTRAN programs, Software - Practice and Experience 1,2, 1971.
[Kn73] D.E. Knuth, Fig.41: Algorithm for "MOVE CORRESPONDING", in 'The art of computer programming vol. 1', §2.4, Addison-Wesley, 1973.