

1 M-1

再帰関数への変換による不変表明の生成

○後藤史博, 西谷泰昭 (群馬大学工学部)

1. はじめに

プログラムの正当性の検証に関してループに対する不変表明を生成することを考える。この不変表明の生成に関しては、Katz と Manna[1], Wegbreit[2], Dunlop と Basili[3], 淡等 [4] などの方法がある。これらの方法は、フローチャートプログラム上または等価な論理プログラム上で不変表明を生成した。

ここで提案する方法は、フローチャートプログラムでの繰り返し部分を等価な再帰関数で置き換えることにより、繰り返しのない単純なフローチャートプログラムに変換する。そしてプログラムの検証を変換した再帰関数に関する帰納法により証明を試みる。通常、この証明はうまくいかないが、以下で述べる条件のもとで証明すべき式として、再帰関数において常に成り立つ関係式を導くことができる。この関係を一般化し、再帰関数を使わずに表したものを不変表明の候補とする。再帰関数についての関係式を求めることができるのは、postcondition にある論理式や関数の定義 (以下基本定義) とプログラムとの構造が一致している場合である。

2. 繰り返しプログラムと再帰関数の検証

フローチャートプログラムの繰り返し部分を等価な再帰関数に書き換えることにより、プログラムを繰り返しのない単純なフローチャートで書き表す。このときの繰り返しにおける検証と再帰関数に対する検証との関係を以下に述べる。

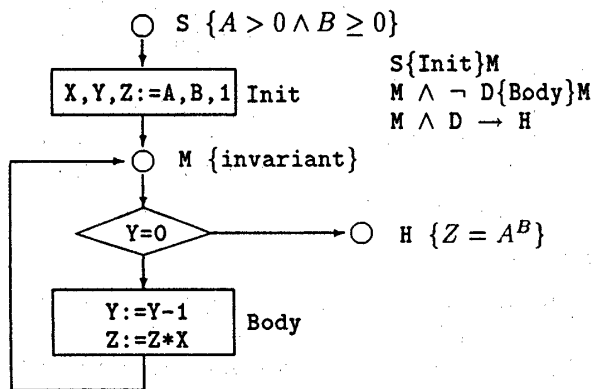
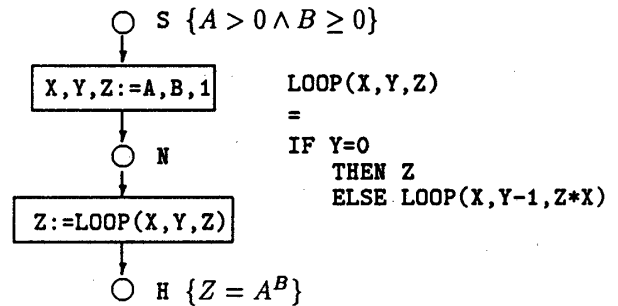
図1. A^B を計算する繰り返しプログラムと検証

図1の検証は、S-M間、M-M間、M-H間のそれぞれが正当であればS-H間において正当である。それに対し、

Derivation of invariant using inductive proof
Fumihito GOTOH, Yasuaki NISHITANI
GUNMA Univ.

繰り返し部分を再帰関数 LOOP で等価変換したフローチャートプログラムを図2に示す。

図2. A^B を計算する再帰プログラムと再帰関数

このときの検証はS-N間、N-H間の検証にわけられる。N-H間の $N \rightarrow wp(\text{LOOP}, H)$ の帰納法による証明は、

$$\text{Base case} : N \wedge D \rightarrow H \quad (1)$$

$$\text{Induction step} : N \wedge \neg N' \wedge \neg D \rightarrow H' \quad (2)$$

となる。ただし'のついたものは、変数をループのBodyに従い変換したものである。図2のBase caseは、 $y = 0 \wedge x > 0 \wedge y \geq 0 \rightarrow \text{LOOP}(x, y, 1) = x^y$ であり、Induction stepは、次のようになる。

$$y \neq 0 \wedge x > 0 \wedge y \geq 0 \\ \wedge (x \leq 0 \vee y - 1 < 0 \vee \text{LOOP}(x, y - 1, 1) = x^{y-1}) \\ \rightarrow \text{LOOP}(x, y, 1) = x^y$$

$N=M$ とし、繰り返しプログラムと再帰関数の検証を比較するとS-M間は共通であり、M-H間は(1)式と同じである。また繰り返しプログラムにおけるM-M間では、 $M \wedge \neg D \rightarrow M'$ の証明であり、これは、(2)式の左辺の否定である。したがって図1のプログラムが正当であれば図2のプログラム検証も同様に行なうことができる。

今、NをS-N間でのstrongest postconditionであるとすると、 $N \rightarrow M$ であり、N-H間での検証である $N \rightarrow wp(\text{LOOP}, H)$ の証明は、 $N=M$ とした時に真であれば必ず真となる。しかし、帰納法による証明において機械的に証明できない再帰関数に関する論理式が現れる。この論理式が真でなければならないということから、この論理式は再帰関数 LOOP が持つ常に成立する関係式であると考えられる。

3. 不変表明の生成法と関係式の一般化

再帰プログラムの検証において先に述べたように、 $N \rightarrow wp(\text{LOOP}, H)$ から再帰関数 LOOP に対する関係を導き、再帰関数 LOOP を使わず、定義だけにより関係を表したものを不変表明とする。その手順を以下に示す。

1. $N \rightarrow wp(\text{LOOP}, H)$ を簡単化、等式の利用等により、単純な論理式にする。
2. LOOP に関する帰納法により証明を行ない、LOOP とその中で使う関数だけの関係式を作る。このとき、プログラムの構造と基本定義の構造の一致が必要。
3. LOOP とその中で使う関数に対し引数の対応をとる。
4. 変数の置き換えにより、再度 LOOP に関する帰納法による証明を行ない関係式を作る。
5. 引数の対応をとり、3 でとった引数の対応とを考慮合わせ一般的な変数に置き換える。
6. 5 で一般化した関係式を証明の過程で作られた論理式を使い、定義だけで表される関係式に書き換える。

以上の操作により作られた関係式を、繰り返しプログラムにおける不変表明の候補とする。1-4 については Boyer Moore の定理証明器 [5] に基づいた証明を試みる。

図 1、図 2 に示した A^B を計算するプログラムの不変表明の生成例をあげる。1 により、

$$x > 0 \wedge y \geq 0 \rightarrow \text{LOOP}(x, y, 1) = x^y$$

2 により、(帰納法は [5] のメジャーセットに基づく)

$$x > 0 \wedge y > 0 \wedge \text{LOOP}(x, y-1, 1) = x^{y-1}$$

$$\rightarrow \text{LOOP}(x, y-1, x) = x^y$$

$x^y = x * x^{y-1}$ であることを知っていれば x^y を展開し、

$$x > 0 \wedge y > 0$$

$$\rightarrow \text{LOOP}(x, y-1, x) = x * \text{LOOP}(x, y-1, 1)$$

3 により、

	左辺	右辺
LOOP の第 1 引数	x	x
LOOP の第 2 引数	y-1	y-1
LOOP の第 3 引数	x	1
* の引数	1	x
* の引数	LOOP	LOOP

4 により、x を X、y-1 を Y で置き換え、

$$X > 0 \wedge Y > 0$$

$$\rightarrow \text{LOOP}(X, Y-1, X^2) = X^2 * \text{LOOP}(X, Y-1, 1)$$

5 により、

	左辺	右辺
LOOP の第 1 引数	X	X
LOOP の第 2 引数	Y-1	Y-1
LOOP の第 3 引数	X^2	1
* の引数	1	X^2
* の引数	LOOP	LOOP

一般化した関係式は

$$X > 0 \wedge Y \geq 0$$

$$\rightarrow \text{LOOP}(X, Y, Z) = Z * \text{LOOP}(X, Y, 1)$$

6 により、プログラムの不変表明は、

$$X > 0 \wedge Y \geq 0 \rightarrow A^B = Z * X^Y$$

となる。

4. プログラムと基本定義の構造の一致

前節の方法は、繰り返し部分を書き換えた再帰関数に関する帰納法により証明を試み、その結果、再帰関数が常に持つ関係が、基本定義によって表される。そのとき、基本定義がプログラムの構造と一致していれば、基本定義の展開により再帰関数がつねに持つ関係を基本定義を使わずに表すことができ、後の一般化を行なう関係を作ることができる。しかし、プログラムの構造と基本定義の構造が一致していないと、基本定義を展開しても一般化が行なえる関係が作れない。例えば、 $x^{2^y} = (x^2)^y$ という関係に基づき A^B を計算するプログラムの不変表明の生成において、次の論理式が現れる。

$$\text{ODD}(y) \wedge \text{LOOP}(x^2, \frac{y-1}{2}, 1) = (x^2)^{\frac{y-1}{2}}$$

$$\rightarrow \text{LOOP}(x^2, \frac{y-1}{2}, x) = x^y$$

この論理式は y が奇数のとき、 $x^y = x * (x^2)^{\frac{y-1}{2}}$ であることを知っていれば、 x^y の展開により

$$\text{ODD}(y)$$

$$\rightarrow \text{LOOP}(x^2, \frac{y-1}{2}, x) = x * \text{LOOP}(x^2, \frac{y-1}{2}, 1)$$

という再帰関数 LOOP の関係式を表すことができる。しかし、 $x^y = x * x^{y-1}$ であることしか知らなければ最初の式のまま何もできない。したがって基本定義は多種の異なる構造を持つことが必要となる。もしくは、1 つの単純な構造を持つ基本定義を異なる構造に変換する機能が必要となる。

この定義の構造について、これまでの不変表明生成法では、人間が知識として持っていることとして必要な時に適した定義の構造を使用している。したがって、定義の構造を変換するヒューリスティックが数多く求められる。また、この部分が不変表明の生成、さらにはプログラムの自動生成のキーになると考えられる。

5. まとめ

繰り返しプログラムを再帰プログラムに変換することで、再帰関数を持ついつでも成立する関係式を導き、関係式を定義により表すことで不変表明を生成した。

現在は Boyer Moore 型の定理証明器に従い手計算による試行を行なっている。

今後の課題としては、理論的な枠組を考察することと 1 つの単純な基本定義の構造をプログラムの構造に一致させるように変換するヒューリスティックを考え、不変表明を生成するシステムを作成することである。

参考文献

- [1] S. Katz and Z. Manna : "Logical analysis of programs", CACM, 19, 4, pp. 188-206 (1976).
- [2] B. Wegbreit : "The synthesis of loop predicates", CACM, 17, 2, pp. 102-112 (1974).
- [3] D. D. Dunlop and V. R. Basili : "A heuristic for driving loop function", IEEE Trans. Software Eng., SE-10, 3, pp. 275-285 (1984).
- [4] 淡、山口、角所、手塚 : "等価論理プログラムに基づく不変表明の帰納的生成法", 信学論 (D), J69-D, 5, pp. 706-713 (昭 61).
- [5] R. S. Boyer and J. S. Moore : "A Computational Logic", Academic Press, (1979).