高階関数型言語におけるランキング関数の回帰推定

村本 大起[1,a)]　佐藤 亮介[1,b)]　鵜林 尚靖[1,c)]　亀井 靖高[1,d)]

概要：ソフトウェアの停止性検証は，プログラム理解において重要な問題である．停止性検証を行うためには，ランキング関数と呼ばれる，ループまたは再帰ごとに値が常に減少し，かつ最小値を抑えられる算術式を推定することが必要である．高階関数型言語において停止性検証を行う既存研究では，末尾再帰関数のような引数が多いものに関しては正確な推定が難しく，検証できないものが多く存在している．よって本研究では，線形回帰を用いてランキング関数の推定を行うことで，引数の多いプログラムについても検証ができる様に改善を試みる．

1. はじめに

ソフトウェアの停止性を検証することは，プログラムの安全性 (safety) や活性 (liveness) の特性を得るために重要であり，プログラムを理解する上では重要な課題となる．

ソフトウェアの停止性を検証する方法としては，ランキング関数を発見してその妥当性を検証することが一般的である．ここでランキング関数とは，プログラムのループまたは再帰ごとに常に減少し続ける，すなわち単調減少する算術式であり常に 0 以上となる算術式のことである．例えば，以下の階乗を求めるプログラム fac を考える．

```
fac n = if n<2 then 1 else n*fac (n-1)
main = fac *int
```

この場合 n は再帰ごとに 1 ずつ減少し，1 以下になった時にプログラムは終了するため n は 1 以下にはならず，ランキング関数 $r(\tilde{x})$ は $n-1$ となる．このようなループ，再帰の中で常に成り立つ関係のことを不変条件という．一つ再帰した後の n を n'，ここでは，$n' = n-1$ として，不変条件 $r(n) > r(n') \wedge r(n) \geq 0 \Leftrightarrow n-1 > n'-1 \wedge n-1 \geq 0$ が得られる．不変条件を満たす $r(\tilde{x})$ が得られれば，その値は単調減少するため，一定値以下にはならない性質によりそのプログラムは停止することがわかる．

高階関数型言語に対する既存研究 [1] では，プログラムの構造からそれぞれの時点での変数に対してランキング関数となりうる条件を洗練することにより推定を行なっている．しかし，変数が増えた場合などではランキング関数を推定することができないことがある．例えば，以下のような階乗を求める末尾再帰型のプログラム fac_acc を考える．

```
fac_acc n acc = if n<2 then acc
    else fac_acc (n-1) (n*acc)
main = fac_acc *int 1
```

末尾再帰のプログラムでは計算結果を渡す引数が追加されているため，推定の際に条件が多くなり失敗することが多い．実際，既存手法 [1] では fac に対してはランキング関数を推定できるが，fac_acc に対しては推定することができない．同様にプログラムの挙動に直接関係のない引数を含むプログラムに関しては推定できないものが多く存在する．

そこで本研究では，手続き言語のテストに基づくランキング関数を線形回帰で推定する研究 [2] に着目し，学習データの取り方を変更することで関数型言語に対しても同様に適用できると考え適用を試みた．線形回帰を用いることで挙動に関係のない引数を含むプログラムに関しても推定できるものが増えると考えられる．実際に fac_acc に対してランキング関数を推定できることを確認した．

2. 対象言語とプログラムの停止性

本節では対象とする高階関数型言語，停止性を定義する．

2.1 対象言語

プログラムを P，式を e，値を v として，対象とする高階関数型言語 L を以下のように定義する．ここで，$\tilde{x}$ は空ではない変数列 $x_1 \cdots x_n$ を表し，$|\tilde{x}|$ は $\tilde{x}$ の長さを表す．$arity(f_i)$ は関数 f_i の引数の数を表す．L は定数 c として $true, false$, 整数値を持ち，$\rightarrow_P$ はプログラム P の環境での 1 ステップ評価を表す．式の定義において，$*_{int}$ は非決定的にある整数に評価される．

[1] 九州大学
[a)] muramoto@posl.ait.kyushu-u.ac.jp
[b)] sato@ait.kyushu-u.ac.jp
[c)] ubayashi@ait.kyushu-u.ac.jp
[d)] kamei@ait.kyushu-u.ac.jp

$$\text{プログラム } P ::= f_1\tilde{x}_1 = e_1, \ldots, f_n\tilde{x}_n = e_n$$
$$\text{式 } e ::= v \mid x \mid \texttt{let}\, x = e_1\, \texttt{in}\, e_2 \mid e_1\, op\, e_2 \mid e_1\, e_2$$
$$\mid \texttt{if}\, e_1\, \texttt{then}\, e_2\, \texttt{else}\, e_3 \mid *_{int}$$
$$\text{値 } v ::= c \mid f \mid f\tilde{v} \quad (|\tilde{v}| < arity(f))$$

2.2 停止性

プログラムの停止を形式的に定義する．

定義 1．

プログラム P が停止するとは，$main \rightarrow_P e_1 \rightarrow_P e_2 \rightarrow_P \cdots$ という無限の簡約列が存在しないときである．

3. 提案手法

本節では停止性の検証方法を述べ，今回適用したランキング関数の推定方法について述べる．

3.1 不変条件による停止性検証

不変条件を満たすランキング関数 $r(\tilde{x})$ の推定によって停止性検証を行う．全体のフローを図 1 に示す．まず，入力として P が与えられた場合，$r(\tilde{x})$ として適当な値 (例えば，0) を与える．そして，既存研究 [1] で用いられていた検証器で判定する．検証器では，ランキング関数を与えると，$r(\tilde{x}) > r(\tilde{x'}) \wedge r(\tilde{x}) \geq c$ を不変条件として妥当性チェックを行う．[*1]チェックに成功した場合，$r(\tilde{x})$ を最終的なランキング関数として，プログラム P は停止するとして結論づける．失敗した場合，失敗する反例を基に再度 $r(\tilde{x})$ を推定して，検証器の $r(\tilde{x})$ を新たな $r(\tilde{x})$ に置き換えて妥当性チェックを行う．ここで，既に反例が出たものと同じ推定しかできず，推定に失敗するような場合は，停止しないかもしれないと結論づけて終了する．[*2]

3.2 ランキング関数の推定

Nori ら [2] は，テストを作成するという観点から変数の値をデータとした線形回帰を用いて，ランキング関数を推定している．本研究では関数型言語に対して反例の値の集合から線形回帰によってランキング関数の推定を行う．

学習データとして反例を用いる．反例は，現在のランキング関数が妥当でない時の変数の組で，関数呼び出しの深さ (再帰での呼び出し数) と全ての変数の値，加えて，Nori ら [2] と同様に深さと変数の変動の指標とするためにそれぞれの変数の初期値を含む．[*3]回帰モデルとしては，$r(\tilde{x}) = a_0 + a_1x_1 + \cdots + a_kx_k$ というテンプレートを用いる．深さを目的変数，その他の値を説明変数として二乗誤差を最小にするような $a_0, a_1, \cdots, a_k$ の推定を行い，推定した $a_1 \cdots a_k$ を元に c を下回らないように a_0 を調整する．

$$\min\ \Sigma_i(a_{0i} + a_{1i}x_1 + \cdots + a_{ki}x_k)^2$$
$$a_0 + a_1x_1 + \cdots + a_kx_k \geq c$$

不変条件についての妥当性チェックに失敗した場合は，反例を学習データに追加して再度推定を行う．

例として，fac_acc について考える．まず，$r(\tilde{x}) = 0$ を与えると，$n = 2$ において単調減少ではないため不変条件を満たさない．そこで，表 1 のような変数の値を反例として学習データに追加して新たに $r(\tilde{x})$ を推定する．ここでは，$r(\tilde{x}) = n - acc + 1$ が得られる．チェックが失敗する限り同様の操作を行い，$r(\tilde{x}) = n - n_{init} - 1$ が得られる．

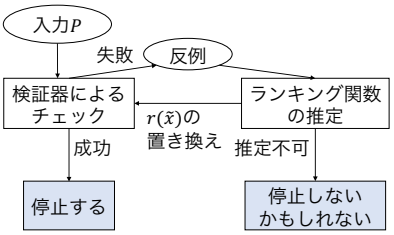

図 1　全体のフロー

表 1　$n = 2$ の時の反例データ (深さと変数とその初期値)

	c	$x_1 = n$	$x_2 = n_{init}$	$x_3 = acc$	$x_4 = acc_{init}$
$\tilde{x}$	1	2	2	1	1
$\tilde{x'}$	2	1	2	2	1

4. まとめと今後の課題

本研究では，既存の関数型言語に対するランキング関数の推定を改良し，より多くのプログラムに対して停止性検証を行うことを目的とした．プログラムの変数と深さから線形一次モデルの線形回帰でランキング関数の推定を行うことで，既存手法では停止性を検証することができないものの検証を行うことができるようになると考えられる．

今回の手法では，アッカーマン関数などランキング関数を線形式で表せないものに関しては推定が難しいため，推定の方法の変更または拡張を考える必要がある．

謝辞

本研究の一部は JSPS 科研費 JP26240007 による助成を受けたものである．

参考文献

[1] Kuwahara, T., Terauchi, T., Unno, H. and Kobayashi, N.: Automatic Termination Verification for Higher-Order Functional Programs, *ESOP2014*, pp. 392–411 (2014).
[2] Nori, A. V. and Sharma, R.: Termination proofs from tests, *ESEC/FSE'13*, pp. 246–256 (2013).

*1 $\tilde{x'}$ は $\tilde{x}$ より 1 つ以上深い再帰 (後の呼び出し) の変数列を表す．
*2 ランキング関数の推定は決定不能問題のため，完全な結論は出すことはできない
*3 変数の初期値を $\tilde{x}_{init}$ とする