数理解析研究所講究録
1093 巻 1999 年 148-154

代数的仕様における振舞等価性証明のための線形文脈帰納法について

二井靖彦　　坂部俊樹
Yasuhiko Nii　　Toshiki Sakabe
名古屋大学大学院 工学研究科 情報工学専攻
464-8603 名古屋市千種区不老町
nii@sakabe.nuie.nagoya-u.ac.jp
sakabe@nuie.nagoya-u.ac.jp

概要

等式の両辺が振舞等価であることを証明することは、代数的仕様の実現検証において重要な役割を果たす。本稿では、線形文脈という限られた形の文脈の長さに関する帰納法による振舞等価性証明法を提案し、その有効性をいくつかの例で示すとともに、Coinduction 法との関係について考察する。

キーワード: 代数的仕様、振舞等価性、文脈帰納法

1 はじめに

代数的仕様の実現検証において、実現の正しさとは与えられた仕様と実現を記述した仕様 (実現仕様) が振舞的に区別がつかないときであるとされる。この振舞的な正しさの関係は振舞実現関係 [1, 2] といわれる。

振舞実現関係を検証するためには、与えられた仕様の各公理の両辺が実現仕様のもとで振舞的に等価であることを証明すれば十分である。2つの式が振舞的に等価であるというのは観測可能な結果を返すすべてのテストによってそれらを区別できないことをいう [2]。したがって振舞等価性を証明するためには、あらゆる観測可能な文脈で表される、すべてのテストに対して等価性を調べる必要がある。しかし、無限に存在する観測可能な文脈すべてについて等価性を調べることは、一般には容易ではない。

振舞等価性を検証する手法の1つに Coinduction 法 [4] があるが、これは隠蔽合同 [4] という概念を用いることにより、振舞等価を間接的に扱い証明をしやすくする手法である。しかし隠蔽合同の発見は人間の力に依るところが大きく、自動化しにくいという欠点がある。

本研究では線形文脈という限られた形の文脈に対して等価性を調べることにより、振舞等価性を検証する手法を提案する。線形文脈とは、文脈の根から穴までの経路以外はすべて変数であるような文脈である。このような文脈の根から穴までの長さに関する帰納法が線形文脈帰納法である。

本稿では、線形文脈帰納法を用いた振舞等価性の検証法を提案し、線形文脈帰納法が健全であることを示す。また、線形文脈帰納法を用いた検証法の有効性をいくつかの例を用いて示すとともに、線形文脈帰納法と Coinduction 法との間の関係について考察する。さらに、Coinduction 法の発見的な部分を解消するテスト集合 Coinduction 法 [5, 6] との比較も行う。

2 振舞等価性と振舞実現関係

振舞的な枠組みにおける代数的仕様を振舞仕様といい、振舞仕様 $SP = (\Sigma, V, E)$ は、シグニチャ $\Sigma = (S, F)$ ($S = V \cup H$, V: 観測可能ソートの集合、H: 隠蔽ソートの集合)、公理の集合 E からなる。

与えられた振舞仕様に対して、それを実現する仕様が正しいということ、すなわち、両者は振舞実現の関係にあるということを検証することは、与えられた仕様の公理が実現のもとで振舞的に等価であるかどうかを検証することである。

まず最初に、振舞等価性に必要な観測可能文脈を定義する。

定義 1 (観測可能文脈 [3, 4])

観測可能文脈は観測可能ソートの項であり、それは変数に加えて文脈の穴を表す隠蔽ソート ($s \in H$) の変数 (文脈変数 z_s または []) のちょうど1回の出現を含む項からなり、$c[z_s]$ または $c[]$ で表し、観測可能文脈の集

合を C_Σ^{obs} と表す。また、文脈 $c[z_s]$ の中の z_s を項 $t \in T(\Sigma, X)$ で置き換えて得られる項を $c[t]$ と表す。

この観測可能文脈を用いて2つの要素の間の振舞等価性を次のように定義する。

定義 2 (振舞等価性 [3, 4])
次の条件を満たす代数 A 上の関係 $\approx$ は振舞等価関係であり、a と a' は振舞等価であるという。ここで、$a, a' \in A$ とし、$\mathrm{sort}(a)$ を a のソートを表すとする。また、V を観測可能ソートの集合、H を隠蔽ソートの集合とする。

1. $\mathrm{sort}(a) = \mathrm{sort}(a') = v \in V$ のとき、
   $a \approx_v a' \Leftrightarrow a = a'$
2. $\mathrm{sort}(a) = \mathrm{sort}(a') = h \in H$ のとき、
   $a \approx_h a' \Leftrightarrow \forall c[] \in C_\Sigma^{obs}, c[a] = c[a']$

以上から、振舞実現は次のように定義される。

定義 3 (振舞実現 [3])
SP=(Σ, V, E)、SPI=$(\Sigma I, VI, EI)(\Sigma \subseteq \Sigma I)$ を振舞仕様、E、EI をそれぞれ仕様 SP、SPI の公理の集合とする。また、$V \subseteq VI$ を観測可能ソートの集合とする。

仕様 SP のすべての公理を振舞的に満たす代数を SP の振舞モデルといい、SP の振舞モデルの集合を $Beh(SP)$ とかく。また、$Beh(SPI) \neq \emptyset$ とする。

このとき、$\forall B \in Beh(SPI), B|_\Sigma \in Beh(SP)$ が成り立つならば、SPI は SP の振舞実現であるという。

振舞実現を形式的に定義するとこのようになるが、このままでは検証がしにくい。そこで、振舞実現に関する次の定理が [3] で証明されている。

定理 4 [3] SP=(Σ, E)、SPI を代数的仕様、E を仕様 SP の公理の集合とする。ここで、σ を基礎代入とする。

仕様 SPI は仕様 SP の振舞実現である。

$$\Updownarrow$$

$\forall (l = r) \in E, \forall c[] \in C_\Sigma^{obs}, \forall \sigma,$
$SPI \vdash c[\sigma(l)] = c[\sigma(r)] \cdots (*)$ が成り立つ。

この定理により、式 (*) が成り立つことがいえれば振舞実現関係を証明できる。

3 線形文脈帰納法

定理4により、(*) が成り立つことを証明すれば振舞実現を検証できるが、これを証明するには、無限に存在する観測可能な文脈をすべて用いなければならない。そこで、我々は無限に存在するすべての文脈を用いるかわりに、線形文脈という文脈を用いて証明する手法を提案する。線形文脈とは、文脈の根から穴までのノードには関数記号が入り、それ以外の葉のところにはすべて変数が入る様な形をしている文脈である。

定義 5 (線形文脈〈観測可能線形文脈〉)
線形文脈は次のように定義される文脈である。ここで、f : 隠蔽ソートの関数、δ : 観測可能ソートの関数とする。

1. $[]$ (隠蔽ソートの文脈変数) は線形文脈である。
2. $c[]$ が線形文脈のとき、
   $sort(c[]) = sort(x_i)$ ならば
   $f(x_1, \cdots, x_{i-1}, c[], x_{i+1}, \cdots x_n)$ は線形文脈である。(ここで、$x_i (1 \leq i \leq n)$ は新しい変数。)

上の 1、2 を繰り返して構成される文脈が線形文脈であり、線形文脈の集合を C_Σ^{lin} と表す。

また、$c[]$ が線形文脈で $sort(c[]) = sort(y_j)$ のとき、$\delta(y_1, \cdots, y_{j-1}, c[], y_{j+1}, \cdots, y_n)$ を観測可能な線形文脈という。(ここで、$y_j (1 \leq j \leq m)$ は新しい変数。)

上で定義された線形文脈 $c[]$ の、根から穴までの文脈の長さ $|c|$ に関する帰納法が、線形文脈帰納法である。形式的には、SPI= $(\Sigma I, EI)$ が SP= $(\Sigma, E)(\Sigma \subseteq \Sigma I)$ の振舞実現であることの証明の概略は次のようになる。ここで、$c[], c'[] \in C_\Sigma^{lin}$、$\delta \in \Sigma$ を観測可能関数、$d_1, \cdots, d_n$ を δ の引数とする。

証明の概略: E のすべての等式 $(l = r)$ に対して、以下のことを証明する。

- 基底段階: 線形文脈の根から $[]$ までの長さ $|c| = 0$ のとき、すなわち、線形文脈が $[]$ のとき、
  $sort([]) = sort(d_i)$ ならば
  $SPI \vdash \delta(d_1, \cdots, d_{i-1}, l, d_{i+1}, \cdots, d_n) = \delta(d_1, \cdots, d_{i-1}, r, d_{i+1}, \cdots, d_n)$
  が成り立つことを証明する。

- 帰納段階: 線形文脈の根から $[]$ までの長さ $|c|$ が n 以下のとき成り立つと仮定する。
  $SPI \vdash \delta(d_1, \cdots, d_{i-1}, c[l], d_{i+1}, \cdots, d_n) = \delta(d_1, \cdots, d_{i-1}, c[r], d_{i+1}, \cdots, d_n) \cdots$ I.H.
  $|c'| = n + 1$ のとき、$sort(c'[]) = sort(d_i)$ ならば
  $SPI \vdash \delta(d_1, \cdots, d_{i-1}, c'[l], d_{i+1}, \cdots, d_n) = \delta(d_1, \cdots, d_{i-1}, c'[r], d_{i+1}, \cdots, d_n)$
  が成り立つことを証明する。

以上の2つ段階における命題を、仕様 SPI の等式を用いた書き換えのみで証明する。

補題 6
Σ をシグニチャ、g を観測可能関数、$c[]$ を根から穴までのノードに観測可能関数が *1* つも含まれない文脈とする。このとき、$g(\cdots, c[], \cdots)$ で表されるような文脈の集合を D で表す。

集合 D の文脈と観測可能文脈について次のことが成り立つ。ここで、$\mathcal{B}$ は Σ 代数、$l, r \in T(\Sigma, X)$ とする。

$\forall c \in D, \mathcal{B} \models c[l] = c[r] \Leftrightarrow \forall c \in C_\Sigma^{obs}, \mathcal{B} \models c[l] = c[r]$

証明 $\Leftarrow$については明らか。よって、$\Rightarrow$について示す。

定義より、$D \subset C_\Sigma^{obs}$。これと仮定より、$c \in (C_\Sigma^{obs} \cap D)$であるような文脈については明らかに成り立つ。よって、$c \in (C_\Sigma^{obs} - D)$について示す。このとき、次のことがいえる。

$$\forall c \in (C_\Sigma^{obs} - D), \exists c' \in D, \exists p, c[]|_p = c'[] \qquad (1)$$

ここで、$c[]|_p$は文脈$c[]$の位置pにおける部分文脈とする。このとき、$c[]$を$c''[c'[]]$と表すとする。

仮定より、$\mathcal{B} \models c'[l] = c'[r]$であるから、両辺に同じ文脈$c''[]$を被せても成り立つ。よって、
$\mathcal{B} \models c''[c'[l]] = c''[c[r]]$が成り立ち、$\Rightarrow$についても成り立つことが言える。以上から補題は証明された。

次に、集合Dに含まれる文脈の線形化に関する次の手続きを示す。

定義 7 (集合 D の文脈の線形化)

文脈$c[] \in D$は次の手続きで線形化される。
ここで、$c[] \equiv f(t_1, \cdots, c'[], \cdots, t_n)$とし、
$t_i \in T(\Sigma, X)(1 \leq i \leq n)$, f:観測可能関数とする。

1. $c'[]$が線形ならば各項t_iをそれぞれ新しい変数x_iで置き換える。
2. $c'[]$が線形でないならば、$c'[]$を線形化して*1*に戻る。

次に線形文脈帰納法に関する次の定理を示す。

定理 8 (線形文脈帰納法による証明法の健全性)

$$\forall (l = r) \in E, \forall c' \in C_\Sigma^{lin}, SPI \models c'[l] = c'[r]$$
$$\Downarrow$$
$$\forall (l = r) \in E, \forall c \in C_\Sigma^{obs}, SPI \models c[l] = c[r]$$

が成り立つ。（ここで、C_Σ^{lin}は観測可能線形文脈の集合とする。）

証明

代数$\mathcal{A}$を仕様SPIのモデルとする。

仮定より、$\mathcal{A} \models c'[l] = c'[r]$がいえるので、次の式が成り立つ。

$$\forall \phi : T(\Sigma, X) \to \mathcal{A}, \phi(c'[l]) = \phi(c'[r])$$

ここで、定義7より集合Dに含まれるすべての文脈$c \in D$に対して次のような代入$\sigma : X \to T(\Sigma, X)$と線形文脈$c' \in C_\Sigma^{lin}$が存在する。

$$c[] = \sigma(c'[])$$

ここで、σは準同型写像より、

$$\begin{aligned} \phi(c[l]) &= \phi(\sigma(c'[l])) = \phi \circ \sigma(c'[l]) \\ &= \phi \circ \sigma(c'[r]) = \phi(\sigma(c'[r])) = \phi(c[r]) \end{aligned}$$

よって、

$$\forall \phi : T(\Sigma, X) \to \mathcal{A}, \phi(c[l]) = \phi(c[r])$$

が成り立つので、次のことが言える。

$$\forall c \in D, \mathcal{A} \models c[l] = c[r] \qquad (2)$$

また式*(2)*と補題*6*より、

$$\forall c \in C_\Sigma^{obs}, \mathcal{A} \models c[l] = c[r]$$

が成り立つ。以上から、定理は証明された。

この定理により線形文脈帰納法で証明が成功すれば、すべての文脈に対する帰納法で証明が成功することがいえる。

線形文脈帰納法は上の証明の概略にしたがって証明が進んでいく。次節では線形文脈帰納法による証明法を例を用いて説明する。

4 線形文脈帰納法を用いた証明例

本節では、線形文脈帰納法による証明の手順を、Δ/Γ完全な仕様とスタックの、2つの例を用いて説明する。

Δ/Γ完全な仕様は、線形文脈帰納法で完全に自動化できる例であり、スタックは、証明の際、補題を必要とするが、その補題も自動的に発見し、証明も自動化が行えるという例である。

4.1 Δ/Γ完全な仕様

定義 9 (Δ/Γ完全 [4])

仕様SPのすべての等式が次のような形をしているとき、SPはΔ/Γ完全であるという。

$$\delta(m(x)) = t$$

ここで、$\delta \in \Delta$:観測可能関数の集合、$m \in \Gamma$:隠蔽関数の集合、tを観測可能な関数のみからなる項とする。

Δ/Γ完全な仕様の例としては、旗の状態を表す Flag の仕様と2つのメモリセルを持った状態の仕様がある。ここでは、仕様がΔ/Γ完全な場合の線形文脈帰納法による振舞等価性証明として、Flag の仕様を例にとって説明する。

例 10 (Flag)

旗の状態を表す仕様記述 FLAG を下に示す。

フラグの仕様

```
th FLAG is sort Flag .
  pr    DATA .
  ops   (up_) (dn_) (rev_) : Flag -> Flag .
  op    up?_ : Flag -> Bool .
  var F : Flag .
  eq    up? up F = true .                (F1)
  eq    up? dn F = false .               (F2)
  eq    up? rev F = not up? F .          (F3)
endth
```

ops,op が関数、*var* が変数とその型の宣言、*(F1)*～*(F3)* が公理を表している。

このとき、仕様 *FLAG* のモデルが次の等式を振舞的に満たすこと、すなわち、仕様 *FLAG* のもとで次の式の両辺が振舞等価であることを証明する。

$$\mathrm{rev}(\mathrm{rev}(\mathrm{f})) = \mathrm{f} \tag{3}$$

証明 式 *(3)* の両辺が振舞等価であること、すなわち定理 *4*, 定理 *8*より

$$\mathrm{FLAG} \vdash \mathrm{up?}(\mathrm{c}[\mathrm{rev}(\mathrm{rev}(\mathrm{f}))]) = \mathrm{up?}(\mathrm{c}[\mathrm{f}])$$

が成り立つことを線形文脈の長さ $|c|$ に関する帰納法で証明する。

- *Base:*$|c| = 0$ のとき、すなわち線形文脈が [] のとき、次式を証明。

$$\mathrm{up?}(\mathrm{rev}(\mathrm{rev}(\mathrm{f})) = \mathrm{up?}(\mathrm{f}) \tag{4}$$

- *Ind.:* $|c| = n$ の時成り立つと仮定する。

$$\mathrm{up?}(\mathrm{c}[\mathrm{rev}(\mathrm{rev}(\mathrm{f}))]) = \mathrm{up?}(\mathrm{c}[\mathrm{f}]) \quad \cdots \text{I.H.}$$

$n + 1$ のとき、線形文脈の根における場合分けにより、次の 3 つの場合、すなわち線形文脈が up(c[])、dn(c[])、rev(c[]) の場合を考えれば十分である。よって以下の式を証明する。

$$\mathrm{up?}(\mathrm{up}(\mathrm{c}[\mathrm{rev}(\mathrm{rev}(\mathrm{f}))])) = \mathrm{up?}(\mathrm{up}(\mathrm{c}[\mathrm{f}])) \tag{5}$$

$$\mathrm{up?}(\mathrm{dn}(\mathrm{c}[\mathrm{rev}(\mathrm{rev}(\mathrm{f}))])) = \mathrm{up?}(\mathrm{dn}(\mathrm{c}[\mathrm{f}])) \tag{6}$$

$$\mathrm{up?}(\mathrm{rev}(\mathrm{c}[\mathrm{rev}(\mathrm{rev}(\mathrm{f}))])) = \mathrm{up?}(\mathrm{rev}(\mathrm{c}[\mathrm{f}])) \tag{7}$$

以上の *4* つの式が成り立つことが証明され、よって 仕様 *FLAG* は式 *(3)* を振舞的に満たすことが示された。

このように Δ/Γ 完全な仕様の場合においては、線形文脈帰納法を用いて証明が可能であり、線形文脈の長さに関する素朴な帰納法のみを用いているので証明の自動化が可能である。

4.2 スタック

ここでは、スタックの仕様と配列とポインタを用いて実現したスタックの仕様が振舞実現の関係にあることを証明する。この例は証明の際 補題を必要とし、その必要な補題が証明の過程で発見できる場合の例である

例 11 (スタック)

スタックの仕様 STACK と、配列とポインタによりスタックを実現した仕様 PTR||ARR を下に示す。

スタックの仕様

```
th STACK is sort Stack .
 pr  DATA .
 op  empty : -> Stack .
 op  push : Nat Stack -> Stack .
 op  top_ : Stack -> Nat .
 op  pop_ : Stack -> Stack .
 var S : Stack .  var N : Nat .
 eq  top push(N,S) = N .                    (S1)
 eq  pop empty = empty .                    (S2)
 eq  pop push(N,S) = S .                    (S3)
endth
```

ポインタと配列によるスタックの仕様

```
th PTR||ARR is sort Stack
 pr  ARR .
 op  _||_ : Nat Arr -> Stack [prec 10] .
 op  empty : -> Stack .
 op  push : Nat Stack -> Stack .
 op  top_ : Stack -> Nat .
 op  pop_ : Stack -> Stack .
 var I N : Nat .  var A : Arr .
 eq  empty = 0 || nil .                     (SI1)
 eq  push(N, I || A) = s I || put(N,I,A).   (SI2)
 eq  top s I || A = A[I]                    (SI3)
 eq  top 0 || A = 0                         (SI4)
 eq  pop s I || A = I || A                  (SI5)
 eq  pop 0 || A = 0 || A                    (SI6)
endth
```

このとき、仕様 PTR||ARR が仕様 STACK を振舞的に実現していることを証明する。

証明　定理 *4*, 定理 *8*より式 *(8)* が成り立つことを線形文脈の長さ $|c|$ に関する帰納法で証明すればよい。ここで、$E = \{(S1), (S2), (S3)\}$ である。

$$\forall (l = r) \in \mathrm{E}, \forall c \in C^{lin}_{\Sigma_{\mathrm{STACK}}},\ (\mathrm{PTR||ARR}) \vdash c[l] = c[r] \tag{8}$$

- 式 *(S1)* に対しては、両辺は観測可能な項であるので次の式を直接示せば良い。

$$\mathrm{top}(\mathrm{push}(\mathrm{n}, \mathrm{i}||\mathrm{a})) = \mathrm{n}$$

- 式 *(S2)* に対する証明。

1. *Base:*$|c| = 0$ のときすなわち線形文脈が [] のとき、次の式を証明。

$$\mathrm{top}(\mathrm{pop}(\mathrm{empty})) = \mathrm{top}(\mathrm{empty})$$

2. *Ind.:*$|c| = n$ の時成り立つと仮定する。

$$\mathrm{top}(\mathrm{c}[\mathrm{pop}(\mathrm{empty})]) = \mathrm{top}(\mathrm{c}[\mathrm{empty}]) \quad \cdots \text{I.H.}$$

線形文脈の長さが $n + 1$ のとき、線形文脈の穴の直前における場合分けにより、線形文脈が c(pop[])、c(push(m, [])) の 2 つの場合を考えれば十分である。よって以下の式を証明する。

top(c[pop(pop(empty))]) = top(c[pop(empty)])
top(c[push(m, pop(empty))])
= top(c[push(m, empty)])

– 式 *(S3)* に対する証明。

1. *Base:*|c| = 0 のときすなわち線形文脈が [] のとき、次の式を証明。ここで $s = i||a$ とする。

   top(pop(push(n,i||a))) = top(i||a)

2. *Ind.:*|c| = n の時成り立つと仮定する。
   top(c[pop(push(n,i||a))]) = top(c[i||a]) ···I.H.

   線形文脈の長さが $n+1$ のとき、線形文脈の穴の直前における場合分けにより、以下の式を証明する。

$$\begin{aligned} &\text{top(c[pop(pop(push(n,i||a)))])} \\ &\quad = \text{top(c[pop(i||a)])} \qquad (9) \\ &\text{top(c[push(m, pop(push(n,i||a)))])} \\ &\quad = \text{top(c[push(m,i||a)])} \end{aligned}$$

ここで式 *(9)* の証明の際、下の補題が必要となる。この補題は式 *(9)* の証明の過程において次の式が成り立ってほしいことから発見される。

top(c[j||put(n, s(j), a)]) = top(c[j||a])

補題 12 次の式の両辺は振舞等価である。

i1||put(n, i2, a) = i1||a if i2 ≥ i1

この補題はあとで紹介する *Coinduction* 法によるスタックの例の証明において使われる補題と同じ補題である。この補題もまた線形文脈帰納法で証明できる。また、振舞等価性の公理化に関する定理が *[1]* で証明されており、この定理は振舞等価なものを公理として用いて良いということを保証しているので、上の証明の中でこの補題を用いることができ、式 *(9)* が証明される。

以上から上のすべての式が証明され、よって、仕様 PTR||ARR は仕様 STACK を振舞的に実現することが証明された。

このように、このスタックの仕様のような例もまた、線形文脈帰納法を用いて振舞実現関係を検証できる。またこれは、検証の際補題が必要となるがその補題もまた証明の流れの中で発見することができ、その補題自身も線形文脈帰納法で証明することができるという例である。したがって自動的な証明が可能な例であるといえる。

5 他の証明法との比較

振舞等価性を検証する手法の 1 つとして Coinduction 法 [4] が知られている。そこで、例を用いて Coinduction 法による証明と線形文脈帰納法による証明過程を比較し、両証明法の間にどのような関係があるか考察する。さらに、テスト集合 Coinduction 法とも比較する。

5.1 Coinduction 法 [4]

直観的には、Coinduction 法は隠蔽合同という概念を用いて振舞等価性を間接的に扱うことにより、検証をしやすくしようとする手法である。

定義 13 (隠蔽合同 [4])
代数 A 上の合同関係 $\cong$ は、任意の観測可能な要素 a, a' にたいして、

$$a \cong_v a' \Leftrightarrow a = a'$$

を満たすとき、隠蔽合同であるという。ここで、$a, a' \in A_v$、v : 観測可能ソートとする。

また、次の定理が [4] で証明されている。

定理 14 [4] 振舞等価関係は、最大の隠蔽合同である。

この定理により、隠蔽合同であるような関係を見つければ、それを用いて振舞等価性を証明できる。

Coinduction 法による振舞等価性証明の概略を以下に示す。つまり、SP の公理 E のすべての等式が、SPI のもとで振舞等価であることを証明する。ここで、SP = (Σ, E) を仕様、SPI を SP を実現する仕様とする。

証明の概略:

1. 隠蔽合同の候補 (SPI 上の) 関係 R を見つけ、それが隠蔽合同であることを証明する。
2. $\forall (l = r) \in E$, l R r であることを証明する。

ここで問題となるのが、人間が隠蔽合同 R をいかにうまく発見するかということである。

5.2 テスト集合 Coinduction 法 [5,6]

テスト集合 Coinduction 法は、Coinduction 法における隠蔽合同 R を自動的に振舞等価として構成する手法である。直観的に言えば、観測可能文脈全体の集合の中から必要不可欠な文脈を選びだし、それから振舞等価を構成するのがテスト集合 Coinduction 法である。

テスト集合 Coinduction 法は次の順にしたがって振舞等価を構成する。詳細は文献 [5, 6] を参照されたい。

- 与えられた仕様の、条件 (Definition71[6]) を満たす等式から文脈書換えシステムを構成する。
- 振舞等価を構成するために必要な文脈をカバーするような集合 (カバー集合) を構成する。
- カバー集合からテスト集合を構成する。
  テスト集合は条件 (Definition74[6]) を満たすカバー集合である。この条件は直観的には、集合の要素の長さに関する条件である。
- テスト集合を元に振舞等価を構成する。

最後に証明したい式がここで求めた振舞等価の関係にあることを、実現仕様の等式を用いて証明する。

5.3 線形文脈帰納法との比較

先ほどと同じ例を用いて Coinduction 法による振舞等価性の証明と線形文脈帰納法による証明とを比較し、また、スタックの例に対してテスト集合 Coinduction 法と線形文脈帰納法を比較する。

まず、最初に Δ/Γ 完全な仕様について比較してみる。Δ/Γ 完全な仕様の場合、Coinduction 法では隠蔽合同 R を自動的に決めることができるので、テスト集合 Coinduction 法を用いる必要がなくなる。

例 10を Coinduction 法で証明すると次のようになる。

例 15 (Δ/Γ 完全な仕様 (Flag))
式 *(1)* の両辺が仕様 *FLAG* のもとで振舞等価であることを *Coinduction* 法により証明する。

- まず、隠蔽合同の候補 関係 R を次のように定義する。ここで、*f,f'* はソート:*Flag* の変数であるとする。

$$f \ \mathrm{R}\ f' \quad = \quad (\mathrm{up?(f)}) == (\mathrm{up?(f')}) \qquad (10)$$

次にこの関係 R が隠蔽合同であることを示すために、$f1\ \mathrm{R}\ f2$ が成り立っているとき、以下の式が成り立つことを証明する。

$$\begin{aligned}
\mathrm{up?(f1)} \quad &= \quad \mathrm{up?(f2)} && (11)\\
\mathrm{up(f1)} \quad &\mathrm{R} \quad \mathrm{up(f2)} \\
&\Big(\quad \Leftrightarrow \quad \mathrm{up?(up(f1))} = \mathrm{up?(up(f2))} \Big) && (12)\\
\mathrm{dn(f1)} \quad &\mathrm{R} \quad \mathrm{dn(f2)} \\
&\Big(\quad \Leftrightarrow \quad \mathrm{up?(dn(f1))} = \mathrm{up?(dn(f2))} \Big) && (13)\\
\mathrm{rev(f1)} \quad &\mathrm{R} \quad \mathrm{rev(f2)} \\
&\Big(\quad \Leftrightarrow \quad \mathrm{up?(rev(f1))} = \mathrm{up?(rev(f2))} \Big) && (14)
\end{aligned}$$

- 次に、式 *(3)* の両辺が ここで定義した R の関係にあることを証明する。すなわち次の関係が成り立つことを証明する。

$$\begin{aligned}
\mathrm{rev(rev(f))} \quad &\mathrm{R} \quad \mathrm{f} \\
&\Big(\quad \Leftrightarrow \quad \mathrm{up?(rev(rev(f)))} = \mathrm{up?(f)} \Big) && (15)
\end{aligned}$$

結果的に以上が成り立つことがいえ、よって、仕様 *FLAG* は式 *(3)* を振舞的に満たすことが示された。

このように、 Δ/Γ 完全な仕様の場合、両証明法とも自動的な証明が可能である。さらに、線形文脈帰納法による証明と比較してみると、上の式 (15) が線形文脈帰納法の基底段階の式 (4) に対応している。また、式 (12) ～ (14) が線形文脈帰納法の帰納段階の式 (5) ～ (7) にそれぞれ対応していることがわかる。このことから、Δ/Γ 完全な仕様の場合振舞実現の線形文脈帰納法による証明と Coinduction 法による証明が自然に対応がとれ、一方の証明スキームから他方の証明スキームを導くことができることがわかる。

次に、証明の際補題を必要とする例である例 11を Coinduction 法とテスト集合 Coinduction 法で証明し、線形文脈帰納法による証明法と比較する。

例 16 (スタック) 例 *11*の仕様 PTR||ARR が 仕様 STACK を振舞的に実現していることを証明する。

Coinduction 法による証明: この例の場合、*Coinduction* 法でも補題 *12*を用いて証明する。しかし *Coinduction* 法ではこの例に適している補題として補題 *12*を人間が発見してやらなければならない。

1. まず、隠蔽合同の候補 関係 R を次のように定義する。ここで、*i,i1,i2* はソート:*Nat* の変数、*a,a1,a2* はソート:*Arr* の変数であるとする。

$$\begin{aligned}
i1\ \mathrm{R}\ i2 \quad &= \quad i1 == i2 \\
(0||a1)\ \mathrm{R}\ (i||a2) \quad &= \quad I == 0 \\
(i||a)\ \mathrm{R}\ (i||a) \quad &= \quad true \\
(s(i1)||a1)\ \mathrm{R}\ (s(i2)||a2) \quad &= \quad (i1 == i2) \\
&\wedge \quad (a1[i1] == a2[i2]) \\
&\wedge \quad ((i1||a1)\ \mathrm{R}\ (i2||a2))
\end{aligned}$$

2. このように関係 R を定めたときに、補題 *12*の両辺が R の関係にあることを証明する。

$$\mathrm{i1||put(n,i2,a)\ R\ i1||a} \quad \text{if } \mathrm{i2} \geq \mathrm{i1}$$

3. この関係 R が隠蔽合同であることを示す。
4. 仕様 STACK の *(S1)* ～ *(S3)* すべての等式の両辺がここで定義した R の関係にあることを証明する。

結果的に以上が成り立つことがいえ、よって、仕様 PTR||ARR が 仕様 STACK を振舞的に実現していることが示された。

テスト集合 Coinduction 法による証明: 次にテスト集合 *Coinduction* 法で振舞等価 R を構成する。

1. 等式 *(S1)(S3)* から文脈書換え規則を作ると次のようになる。このとき、隠蔽ソートの変数を穴と見なして分岐のない列をつくり、文脈として扱う。

$$top\ push(n)[] \rightarrow \phi \qquad pop\ push(n)[] \rightarrow \psi$$

2. カバー集合を構成すると次のような集合になる。

$$\{(top, \{push(n), pop\})\} \qquad (16)$$

3. カバー集合からテスト集合を構成する。この例の場合、テスト集合の条件にあてはまるカバー集合もまた *(16)* の集合となる。

4. 文脈書換え規則により、除去可能片と壁に分ける。この例の場合、除去可能片は$push(n)$であり、壁はpopとなる。
   よって、*(16)*の集合から、除去可能片を取り除いた集合が、カバー集合$\{(top, \{pop\})\}$となり、これから振舞等価を構成すると次の式*(17)*になる。

   $$s\ R\ t\ =\ (top\ s == top\ t) \wedge (pop\ s\ R\ pop\ t) \quad (17)$$

5. この振舞等価Rを用いて、仕様$STACK$のすべての等式の両辺がRの関係にあることを仕様$PTR||ARR$の等式を用いて証明する。
   このとき、振舞等価が帰納的な定義になっているため、帰納法や補題が必要となる。

これがテスト集合*Coinduction*法によるスタックの振舞実現の検証である。

スタックの例の場合、線形文脈帰納法による証明法とCoinduction法による証明法の対応は全くとれないことがわかる。また、両証明法とも補題を必要としているが、線形文脈帰納法による証明法はほとんど自動的に行えるのに対し、Coinduction法による証明法は隠蔽合同と補題をそれぞれ人間が用意してやる必要があることが欠点であるといえる。

また、この例におけるテスト集合Coinduction法による証明法は、自動的に振舞等価を発見できるので、Coinduction法の自動化に役に立つと思われる。しかし、与えられた仕様の両辺が振舞等価であることを証明する際に、帰納法と補題12が必要となる。冗長な文脈は取り除かれているが、線形文脈帰納法による証明法においても、それらの文脈についての証明は単純な方法で証明できるので、両証明法の違いはそれほどない様に思える。また、[6]には、与えられた仕様のすべての等式が、文脈書換えシステムの条件を満たすことを、テスト集合Coinduction法を行える条件としていたが、線形文脈帰納法では仕様の等式にそのような条件はなくても証明が行えるということが、両者の違いである。

6 まとめと今後の課題

本稿では、振舞等価性を証明する手法として線形文脈帰納法を用いる手法を提案し、線形文脈帰納法が健全であることを示した。

さらに、線形文脈帰納法により、全く自動的に証明が行える例として、Flagの仕様を、補題を必要とするがそれも含めて自動的に証明が行える例として、スタックの仕様を示した。よって、これら2つの例に対しては、容易に自動化ができそうである。また、今回は紹介しなかったが、キューの例に対しては、補題を与える必要があるが、線形文脈帰納法により、証明が可能である。しかし、補題を与える必要があるので、証明の自動化については容易ではないと思われる。

他の証明法と比較すると、Coinduction法では人間が隠蔽合同や適当な補題を与えてやらなければならないのに対し、線形文脈帰納法は人間の関与が少ないので自動化向きであると思われる。また、Δ/Γ完全な仕様の場合においては両証明法の対応が自然に取れ、一方から他方の証明スキームを構成できることを示した。

また、テスト集合Coinduction法では、振舞等価を構成するのに必要な文脈を絞り込み、自動的に振舞等価の適切な形を構成できるが、絞り込んだ文脈が無限に存在する場合、帰納法が必要となる。一方、線形文脈帰納法は、簡潔な帰納法による証明が可能なので、両証明法の効率については、大きな差はないように思われる。

また、与えられた仕様の等式に条件が付いているが線形文脈帰納法ではそのような条件を必要としないので、線形文脈帰納法を用いることができる仕様の幅が、テスト集合Coinduction法よりも広いように思われる。

したがって、線形文脈帰納法$\not\subseteq$テスト集合Coinduction法であることはいえるが、線形文脈帰納法$\not\supseteq$テスト集合Coinduction法であるかどうかは、まだわかっていない。

今後の課題としては、本稿で提案した振舞等価性のための線形文脈帰納法の能力の更なる分析をすることや、この証明法をメタ項書換え計算[8]などで記述し、自動化をはかることなどがあげられる。

参考文献

[1] M.Bidoit and R.Hennicker : 'Behavioural theories and the proof of behavioural properties',Theoretical Computer Science,165(1):3-55(1996).

[2] M.Bidoit and R.Hennicker : 'Proving Behavioural Theorems with Standard First-Order Logic',Lecture Notes in Computer Science 850,41-58,Berlin:Springer, 1994.

[3] R.Hennicker : 'Context Induction: A Proof Principle for Behavioural Abstractions and Algebraic Implementations', Formal Aspects of Computing(1991)3: 326-345.

[4] J.Goguen and G.Malcolm : 'A Hidden Agenda',USCD Technical Report CS97-538,1997.

[5] 松本充広, 二木厚吉 :' 振舞意味論に基づく仕様の検証法', 日本ソフトウェア科学会第14回大会論文集,p473-476, 1997.

[6] M.Matsumoto : 'Verification methods for algebraic specifications', Ms thesis,JAIST,1998.

[7] F.Baader and T.Nipkow : 'Term Rewriting and All That', CAMBRIDGE UNIVERSITY PRESS,1998.

[8] 大島 淳史, 坂部 俊樹 : ' 動的弁別ネットを用いたメタ項書換え計算インタプリタの設計', 電子情報通信学会ソフトウェアサイエンス研究会,1999年3月.