修士論文

作用項書換えのための解釈順序

2110100　田中　哲平

主指導教員　廣川　直
審査委員主査　廣川　直
審査委員　小川　瑞史
石井　大輔
緒方　和博

北陸先端科学技術大学院大学
先端科学技術研究科
(情報科学)

令和 5 年 2 月

Abstract

Term rewriting is a computation model that is based on pattern matching with directed equations, called rewrite rules. In this model, we regard sets of rewrite rules, called term rewrite systems, as declared programs. In particular, term rewrite systems over constant symbols and a unique binary function symbol are called *applicative term rewrite systems*.

Consider the following applicative term rewrite system.

$$\begin{array}{rl} \mathsf{add}\ 0\ y \to y & \qquad \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) & \qquad \mathsf{map}\ f\ (:\ x\ \ell) \to :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \\ \mathsf{double}\ x \to \mathsf{add}\ x\ x & \end{array}$$

The term $\mathsf{map}\ \mathsf{double}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil}))$ is rewritten as follows:

$$\begin{array}{rl} & \mathsf{map}\ \mathsf{double}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to & :\ (\mathsf{double}\ 0)\ (\mathsf{map}\ \mathsf{double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to & :\ (\mathsf{add}\ 0\ 0)\ (\mathsf{map}\ \mathsf{double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to & :\ 0\ (\mathsf{map}\ \mathsf{double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to & :\ 0\ (:\ (\mathsf{double}\ (\mathsf{s}\ 0))\ (\mathsf{map}\ \mathsf{double}\ \mathsf{nil})) \\ \to & :\ 0\ (:\ (\mathsf{add}\ (\mathsf{s}\ 0)\ (\mathsf{s}\ 0))\ (\mathsf{map}\ \mathsf{double}\ \mathsf{nil})) \\ \to & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{add}\ 0\ (\mathsf{s}\ 0)))\ (\mathsf{map}\ \mathsf{double}\ \mathsf{nil})) \\ \to & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{s}\ 0))\ (\mathsf{map}\ \mathsf{double}\ \mathsf{nil})) \\ \to & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{s}\ 0))\ \mathsf{nil}) \end{array}$$

The last term cannot be rewritten any more. Such a term is called normal form, that is regarded as a computational result. Applicative Term rewrite systems underlie functional programming languages and automated theorem provers that handle higher-order logic.

Termination is the property that any term can reach normal forms eventually. In order to show termination, *reduction orders* are employed. If all rules orient from

left to right for some reduction order then its system is terminating. For example, suppose that the above ATRS is oriented by some reduction order $>$ as follows:

$$\begin{aligned} \mathsf{add}\ 0\ y &> y & \mathsf{map}\ f\ \mathsf{nil} &> \mathsf{nil} \\ \mathsf{add}\ (\mathsf{s}\ x)\ y &> \mathsf{s}\ (\mathsf{add}\ x\ y) & \mathsf{map}\ f\ (:\ x\ \ell) &> :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \\ \mathsf{double}\ x &> \mathsf{add}\ x\ x \end{aligned}$$

Then we have the system is terminating. Reduction orders are the basis of not only termination proofs, but also automated theorem provers.

However, it is hard for existing reduction orders to orient the direction that variables are duplicated, therefore $\mathsf{map}\ f\ (:\ x\ \ell) > :\ (f\ x)\ (\mathsf{map}\ f\ \ell)$ and $\mathsf{double}\ x > \mathsf{add}\ x\ x$ are difficult. The aim of this research is to find a new reduction order that can handle variable duplication.

In this thesis we introduce the reduction order by combining interpretation order with *uncurrying* (Hirokawa et al. 2013). Uncurrying is a powerful transformation method for termination proof. Here we illustrate our method by using the above example. Our method allows us to uncurry and interpret at the same time. For termination of the original systems, we just have to find an interpretation order $>$ that orient as follows:

$$\begin{aligned} \mathsf{add}_2(0, y) &> y & \mathsf{map}_2(f, \mathsf{nil}) &> \mathsf{nil} \\ \mathsf{add}_2(\mathsf{s}_1(x), y) &> \mathsf{s}_1(\mathsf{add}_2(x, y)) & \mathsf{map}_2(f, :_2(x, \ell)) &> :_2(f\ x, \mathsf{map}_2(f, \ell)) \\ \mathsf{double}_1(x) &> \mathsf{add}_2(x, x) \end{aligned}$$

Actually, a few conditions are added for the algebra, but it is easier to find interpretations than the original one.

In addition to the definition of new interpretation order, we found an automatable termination criterion by using *polynomial interpretation* (Lankford, 1979). Here we list the contributions of the thesis:

- a new reduction order and
- its automation technique.

Experiments on the Termination Problem Database (TPDB) show effectiveness of our new reduction order.

目次

図目次

表目次

第 1 章

はじめに

現代的なプログラミング言語では、高度な制御抽象を行うために高階関数が多用される。例えば、代表的な高階関数である map は関数型プログラミング言語 Haskell で以下のように定義される。

```
map f [] = []
map f (x:xs) = f x : map f xs
```

`map` は関数を表す変数 `f` を引数として受け取る。`map` の振る舞いを例示するため以下の関数 `double` も同時に宣言されているものとする。

```
double x = x + x
```

このとき、`map double (0:1:[])` は次のように計算される。

```
map double (0:1:[]) = double 0 : map double (1:[])
                    = 0 : map double (1:[])
                    = 0 : double 1 : map double []
                    = 0 : 2 : map double []
                    = 0 : 2 : []
```

プログラムの実行に数学的基礎を与えるために計算モデルが用いられる。項書換えは、書換え規則と呼ばれる式変形規則に基づく計算モデルであり、与えられた項を規則に沿って式変形することで計算を行う。このような書換え規則の集合は項書換え系と呼ぶ。項書換えは宣言型言語（ML, Haskell 等）の理論基盤と用いられる。

1.1 研究目的

一階の項書換えでは高階関数を表現することができないため、それを項書換えの中で扱うための枠組みが提案されている。本研究で対象とする作用項書換え系は高階関数を一階の項書換え系で扱う枠組みである。作用項書換え系で用いられる項は作用項と呼ばれ、用いる関数記号を定数と関数適用記号と呼ばれる唯一つの 2 引数関数のみに限定した項である。次の書換え規則から成る項書換え系 $\mathcal{R}_1$ は作用項書換え系である。

$$\begin{array}{rr} \mathsf{add} \circ 0 \circ y \to y & \mathsf{map} \circ f \circ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{add} \circ (\mathsf{s} \circ x) \circ y \to \mathsf{s} \circ (\mathsf{add} \circ x \circ y) & \mathsf{map} \circ f \circ (: \circ x \circ \ell) \to : \circ (f \circ x) \circ (\mathsf{map} \circ f \circ \ell) \\ \mathsf{double} \circ x \to \mathsf{add} \circ x \circ x & \end{array}$$

ただし関数適用記号 $\circ$ を左結合の中置記法で扱っている。作用項を表記する際は $\circ$ を省略し部分項を並列に並べた記法（並置記法）がよく用いられる。例えば $\mathsf{map} \circ f \circ (: \circ x \circ \ell)$ を並置記法で表すと $\mathsf{map}\ f\ (:\ x\ \ell)$ となる。同様に $\mathcal{R}_1$ を並置記法で表したものは以下のようになる。

$$\begin{array}{rr} \mathsf{add}\ 0\ y \to y & \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) & \mathsf{map}\ f\ (:\ x\ \ell) \to :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \\ \mathsf{double}\ x \to \mathsf{add}\ x\ x & \end{array}$$

$\mathcal{R}_1$ の下で項 $\mathsf{map\ double}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil}))$ は以下のように書き換えられる。

$$\begin{aligned} & \mathsf{map\ double}\ (:\ 0\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to\ & :\ (\mathsf{double}\ 0)\ (\mathsf{map\ double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to\ & :\ (\mathsf{add}\ 0\ 0)\ (\mathsf{map\ double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to\ & :\ 0\ (\mathsf{map\ double}\ (:\ (\mathsf{s}\ 0)\ \mathsf{nil})) \\ \to\ & :\ 0\ (:\ (\mathsf{double}\ (\mathsf{s}\ 0))\ (\mathsf{map\ double\ nil})) \\ \to\ & :\ 0\ (:\ (\mathsf{add}\ (\mathsf{s}\ 0)\ (\mathsf{s}\ 0))\ (\mathsf{map\ double\ nil})) \\ \to\ & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{add}\ 0\ (\mathsf{s}\ 0)))\ (\mathsf{map\ double\ nil})) \\ \to\ & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{s}\ 0))\ (\mathsf{map\ double\ nil})) \\ \to\ & :\ 0\ (:\ (\mathsf{s}\ (\mathsf{s}\ 0))\ \mathsf{nil}) \end{aligned}$$

最後の項 $:\ 0\ (:\ (\mathsf{s}\ (\mathsf{s}\ 0))\ \mathsf{nil})$ は $\mathcal{R}_1$ のどの規則を用いても書き換えられず、そのような項は正規形と呼ばれる。正規形は与えられた入力に対する計算結果を表す。作用項書換えは関数型プログラミング言語の計算モデルとなっている。また、定理自動証明系にも作用項

書換えが利用されているものがあり、高階論理に基づく定理自動証明系は SK コンビネータを用いて命題を型無しの作用項に変換し [8]、作用項上の書き換えによって自動証明を実現している。

どのような項も項書換え系 $\mathcal{R}$ による無限の書き換え列を持たないとき、$\mathcal{R}$ は停止性を持つという。上記の $\mathcal{R}_1$ は停止性を持つ作用項書換え系である。停止性を持たない項書換え系の例として、以下のただ一つの規則から成る $\mathcal{R}_2$ を取り上げる。

$$\mathsf{repeat}\ x \to \mathsf{:}\ x\ (\mathsf{repeat}\ x)$$

$\mathsf{repeat}\ 0$ という項から以下の無限列が得られる。

$$\mathsf{repeat}\ 0 \to \mathsf{:}\ 0\ (\mathsf{repeat}\ 0) \to \mathsf{:}\ 0\ (\mathsf{:}\ 0\ (\mathsf{repeat}\ 0)) \to \cdots$$

停止性はプログラムが任意の入力に対して有限の計算ステップで出力を返すことを保証する性質であり、ソフトウェアの信頼性を担保する上で重要である。項書換え系の停止性を証明する代表的な手法に、簡約順序と呼ばれる項上の停止性を保証する順序を用いるものがある。例えば $\mathcal{R}_1$ において、各規則に対して以下を満たす簡約順序 $>$ が存在すれば $\mathcal{R}_1$ は停止性を持つ。

$$\begin{array}{cc} \mathsf{add}\ 0\ y > y & \mathsf{map}\ f\ \mathsf{nil} > \mathsf{nil} \\ \mathsf{add}\ (\mathsf{s}\ x)\ y > \mathsf{s}\ (\mathsf{add}\ x\ y) & \mathsf{map}\ f\ (\mathsf{:}\ x\ \ell) > \mathsf{:}\ (f\ x)\ (\mathsf{map}\ f\ \ell) \\ \mathsf{double}\ x > \mathsf{add}\ x\ x & \end{array}$$

簡約順序は項書換えを用いた定理証明系（E [21], LEO-III [22], Vampire [18], Zipperposition [9] 等）でも用いられており、停止性証明と定理自動証明の基盤を成している。上記の定理証明系の能力は用いる簡約順序に依存するため、より広いクラスで停止性証明可能な簡約順序の研究が行われている。簡約順序を用いた作用項書換え系の停止性自動解析が本研究のテーマである。

1.2 既存手法

項書換え系のための簡約順序としては、多項式解釈順序 [19] や辞書式経路順序 [15], Knuth–Bendix 順序 [16] が知られている。一方、引数の概念を持たない項である作用項に対しては、一階の項のために設計されたこれらの順序が上手く機能しないことが知られている。作用項上の簡約順序の構成は困難であることが知られており、長らく存在しなかったが、2017 年に Becker らが lambda-free Knuth–Bendix 順序 [5] と呼ばれる順序

を発表し、2021 年には Bentkamp が同じアイディアのもと埋め込み経路順序 [6] を考案した。しかし、これら二つの順序は変数が複製される向きに順序付けることができない短所を持ち、例えば $\mathsf{double}\ x > \mathsf{add}\ x\ x$ や $\mathsf{map}\ f\ (:\ x\ \ell) > :\ (f\ x)\ (\mathsf{map}\ f\ \ell)$ のような順序付けを実現できない。この原因は順序が単純化順序と呼ばれる引数が必ず元の項より小さくなる性質を有するためである。作用項書換え系の停止性証明のための強力な変換手法として、uncurry 化 [13] が知られている。一階の項 $\mathsf{add}(\mathsf{s}(x), y)$ に対応する作用項は $\mathsf{add}\ (\mathsf{s}\ x)\ y$ であり、その変換は curry 化と呼ばれる。逆に後者を前者にする変換を uncurry 化と呼ぶ。

1.3 成果

本論文では uncurry 化と解釈順序を組み合わせた、作用項書換え系のための新たな簡約順序を提案する。その順序による停止性証明の自動化を行う。提案手法は uncurry 化と項の代数解釈を同時に行うことができ、例えば $\mathcal{R}_1$ に対しては以下を満たす解釈順序 $>$ を見つければよい。

$$\begin{aligned} \mathsf{add}_2(0, y) &> y & \mathsf{map}_2(f, \mathsf{nil}) &> \mathsf{nil} \\ \mathsf{add}_2(\mathsf{s}_1(x), y) &> \mathsf{s}_1(\mathsf{add}_2(x, y)) & \mathsf{map}_2(f, :_2(x, \ell)) &> :_2(f\ x, \mathsf{map}_2(f, \ell)) \\ \mathsf{double}_1(x) &> \mathsf{add}_2(x, x) \end{aligned}$$

実際には代数に新たな条件が加えられるが、それでも解釈の発見は元の $\mathcal{R}_1$ と比べて容易になる。提案した順序は単純化順序ではないため、変数が複製される向きに対しても順序付けが可能となっている。また、この順序を用いた停止性証明の自動化を行い、実験によりその有効性を示す。以下に本研究の成果を記す。

- 変数複製を伴う順序に対処可能な、新たな解釈順序の定義（3.3 節, 定理 3.28）
- 自動化可能な有向性基準（3.4 節, 定理 3.34 と系 3.35）
- 多項式解釈を用いた自動化の手順（4 章）

本論文は次の構成になっている。2 章では、項書換え系とその性質について述べる。3 章では、新たな解釈順序の定義とそれが簡約順序となることの証明を行う。4 章では、3 章で定義した順序による停止性証明の自動化の手法について述べる。5 章では、4 章による手法を実装したツールで実験を行い、結果について考察を行う。最後に 6 章で本研究の成果と今後の課題についてまとめる。

第 2 章

準備

本章では項書換えにおける基本的な概念と性質を [3] に基づいて書く。なお、本論文では自然数は 0 以上の整数であるとし、その集合を $\mathbb{N}$ と表す。

2.1 抽象書換え系

定義 2.1. 集合 A 上の**二項関係**（binary relation）とは、直積 $A \times A$ の部分集合のことをいう。集合 A と A 上の二項関係 $\to$ の組 $(A, \to)$ を**抽象書換え系**（abstract rewrite system）という。二項関係 $\to$ について、$(x, y) \in \to$ であることを $x \to y$ で表す。

定義 2.2. A 上の二項関係 $\to$ と $\rightsquigarrow$ の**合成**（composition）$\to \cdot \rightsquigarrow$ を以下のように定義する。

$$\to \cdot \rightsquigarrow = \{(a, c) \mid \exists b \in A.\ a \to b \wedge b \rightsquigarrow c\}$$

定義 2.3. $(A, \to)$ を抽象書換え系とする。$n \in \mathbb{N}$ に対し、$\to^n$ を以下のように帰納的に定義する。

$$\to^n = \begin{cases} \{(a, a) \mid a \in A\} & n = 0 \text{ のとき} \\ \to \cdot \to^{n-1} & n \geq 1 \text{ のとき} \end{cases}$$

定義 2.4. $(A, \to)$ を抽象書換え系とする。

- $\to^0 \cap \to = \varnothing$ であるとき、$\to$ は**非反射的**（irreflexive）であるという。
- $\to^2 \subseteq \to$ であるとき、$\to$ は**推移的**（transitive）であるという。
- $\to$ が $a_0 \to a_1 \to a_2 \to \cdots$ なる無限列を持たないとき、$\to$ は**整礎**（well-founded）であるという。

2.2 項書換え系

定義 2.5. 関数記号の集合 $\mathcal{F}$ を**シグネチャ**（signature）という。関数記号は**アリティ**（arity）の情報も持っており、関数記号 f がアリティ $n \in \mathbb{N}$ であることを強調するために $f^{(n)}$ と書く場合がある。また、**変数**（variable）の集合 $\mathcal{V}$ は $\mathcal{F} \cap \mathcal{V} = \varnothing$ を満たす記号の集合である。$\mathcal{F}$ と $\mathcal{V}$ から成る**項**（term）の集合 $\mathcal{T}(\mathcal{F}, \mathcal{V})$ は、以下の両方を満たす最小の集合である。

- $\mathcal{V} \subseteq \mathcal{T}(\mathcal{F}, \mathcal{V})$
- $t_1, \ldots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ かつ $f^{(n)} \in \mathcal{F}$ ならば $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{F}, \mathcal{V})$

項 t に変数が出現しないとき、t を**基底項**（ground term）と呼ぶ。基底項全体の集合 $\mathcal{T}(\mathcal{F}, \varnothing)$ を単に $\mathcal{T}(\mathcal{F})$ と書く場合もある。

定義 2.6. 項 t の**サイズ**（size）$|t|$ は t に出現する変数と関数記号の数を表し、定義は以下のように与えられる。

$$|t| = \begin{cases} 1 & t \in \mathcal{V} \text{ のとき} \\ 1 + \sum_{i=1}^{n} |t_i| & t = f(t_1, \ldots, t_n) \text{ のとき} \end{cases}$$

また、項 t に出現する変数 $x \in \mathcal{V}$ の数を $|t|_x$ と書き、定義は以下のように与えられる。

$$|t|_x = \begin{cases} 1 & t \in \mathcal{V} \text{ かつ } t = x \text{ のとき} \\ 0 & t \in \mathcal{V} \text{ かつ } t \neq x \text{ のとき} \\ \sum_{i=1}^{n} |t_i|_x & t = f(t_1, \ldots, t_n) \text{ のとき} \end{cases}$$

定義 2.7. 代入（substitution）σ は、集合 $\{x \in \mathcal{V} \mid \sigma(x) \neq x\}$ が有限な、変数から項への関数である。代入は以下のように項から項への関数へと拡張できる。

$$t\sigma = \begin{cases} \sigma(t) & t \in \mathcal{V} \text{ のとき} \\ f(t_1\sigma, \ldots, t_n\sigma) & t = f(t_1, \ldots, t_n) \text{ のとき} \end{cases}$$

$t\sigma$ が基底項となるような代入 σ を項 t の**基底代入**（ground substitution）と呼ぶ。

定義 2.8. 2 つの代入 σ, τ の**合成**（composition）$\sigma\tau$ は以下のように定義される代入である。

$$\sigma\tau = \{x \mapsto (x\sigma)\tau \mid x \in \mathcal{V}\}$$

定理 2.9. σ, τ を代入とする。任意の項 t について $t(\sigma\tau) = (t\sigma)\tau$ が成り立つ。

定義 2.10. $\mathcal{F}$ にも $\mathcal{V}$ にも属さない記号 $\Box$ を**穴**（hole）と呼ぶ。穴を唯一つ含む項を**文脈**（context）と呼ぶ。文脈 C と項 t に対し、項 $C[t]$ を以下のように帰納的に定める。

$$C[t] = \begin{cases} t & C = \Box \text{ のとき} \\ f(s_1, \ldots, C'[t], \ldots, s_n) & C = f(s_1, \ldots, C', \ldots, s_n) \text{ のとき} \end{cases}$$

定義 2.11. R を項上の関係とする。

- $s\ R\ t$ ならば任意の代入 σ に対して $s\sigma\ R\ t\sigma$ が成り立つとき、R は**代入について閉じている**（closed under substitution）という。
- $s\ R\ t$ ならば任意の文脈 C に対して $C[s]\ R\ C[t]$ が成り立つとき、R は**文脈について閉じている**（closed under context）という。
- R が代入と文脈について閉じているとき、R は**書換え関係**（rewrite relation）であるという。

定義 2.12. 書換え規則（rewrite rule）は、以下の二つの条件を同時に満たす項の組 (l, r) である。

1. l は変数ではない。
2. r に出現する変数は全て l にも出現する。

以下、本論文では (l, r) が書換え規則であるとき $l \to r$ と書く。書換え規則の集合を**項書換え系**（term rewrite system, TRS）と呼ぶ。

定義 2.13. $\mathcal{R}$ を項書換え系とする。項 s, t に対して二項関係 $s \to_{\mathcal{R}} t$ が成り立つとは、$s = C[l\sigma]$ かつ $t = C[r\sigma]$ を満たす書換え規則 $l \to r$, 代入 σ, 文脈 C が存在することである。$\to_{\mathcal{R}}$ が整礎であるとき、$\mathcal{R}$ は**停止性を持つ**（terminating）という。

2.3 順序と代数

定義 2.14. $>$ を二項関係とする。$>$ が非反射的かつ推移的であるとき、$>$ は**狭義半順序**（strict partial order）であるという。狭義半順序 $>$ が整礎であるとき、$>$ は**整礎順序**（well-founded order）であるという。

定義 2.15. シグネチャ $\mathcal{F}$ 上の**代数**（algebra）$\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}})$ は以下から構成される。

- **台集合**（carrier）A
- 任意の $f^{(n)} \in \mathcal{F}$ に関連付けられた**解釈**（interpretation）$f_{\mathcal{A}} : A^n \to A$

$>$ を A 上の順序とするとき、組 $(\mathcal{A}, >)$ を**順序代数**（ordered algebra）という。

定義 2.16. $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}})$ を代数とする。写像 $\alpha : \mathcal{V} \to A$ を**割り当て**（assignment）と呼び、**付値関数**（valuation function）$[\alpha]$ を項から A への関数とする。項 t に対する**付値**（valuation）$[\alpha](t)$ は以下のように定義される。

$$[\alpha](t) = \begin{cases} \alpha(t) & t \in \mathcal{V} \text{ のとき} \\ f_{\mathcal{A}}([\alpha](t_1), \ldots, [\alpha](t_n)) & t = f(t_1, \ldots, t_n) \text{ のとき} \end{cases}$$

定義 2.17. $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}})$ を代数、$>$ を A 上の順序とする。任意のアリティが 1 以上の関数記号 $f^{(n)} \in \mathcal{F}$ と任意の $a_1, \ldots, a_n, b \in A$ について、$a_i > b$ ならば $f_{\mathcal{A}}(a_1, \ldots, a_i, \ldots, a_n) > f_{\mathcal{A}}(a_1, \ldots, b, \ldots, a_n)$ が成り立つとき、$(\mathcal{A}, >)$ は**単調**（monotone）であるという。更に、$>$ が整礎であれば $(\mathcal{A}, >)$ は**整礎単調**（well-founded monotone）であるという。

定義 2.18. $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}})$ を代数、$>$ を A 上の順序とする。任意のアリティが 1 以上の関数記号 $f^{(n)} \in \mathcal{F}$ と任意の $a_1, \ldots, a_n, b \in A$ について以下の両方が成り立つとき $(\mathcal{A}, >)$ は

2.4 簡約順序

定義 2.19. 項上の狭義半順序 $>$ が書換え関係であるとき、$>$ は**書換え順序**（rewrite order）であるという。書換え順序 $>$ が整礎であるとき、$>$ は**簡約順序**（reduction order）であるという。

定理 2.20. $\mathcal{R}$ を項書換え系とする。以下は同値である。

- $\mathcal{R}$ が停止性を持つ。
- $\to_{\mathcal{R}} \subseteq >$ を満たす簡約順序 $>$ が存在する。

定義 2.21. $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}})$ を代数、$>$ を A 上の順序とする。任意のアリティが 1 以上の関数記号 $f^{(n)} \in \mathcal{F}$ と任意の $a_1, \ldots, a_n, b \in A$ について、$a_i > b$ ならば $f_{\mathcal{A}}(a_1, \ldots, a_i, \ldots, a_n) > f_{\mathcal{A}}(a_1, \ldots, b, \ldots, a_n)$ が成り立つとき $(\mathcal{A}, >)$ は**単調代数**（monotone algebra）であるという。更に $>$ が整礎であるとき $(\mathcal{A}, >)$ は**整礎単調代数**

(well-founded monotone algebra) であるという。

定義 2.22. $(\mathcal{A}, >)$ を代数とする。**解釈順序** (interpretation order) $>_{\mathcal{A}}$ を項上の二項関係として次のように定める。

任意の割り当て α に対して $[\alpha](s) > [\alpha](t)$ が成り立つならば $s >_{\mathcal{A}} t$

定理 2.23. $(\mathcal{A}, >)$ が整礎単調代数ならば $>_{\mathcal{A}}$ は簡約順序である。

定義 2.24. 単調代数 $(\mathcal{A}, >)$ は以下を満たすとき **多項式解釈** (polynomial interpretation) と呼ばれる。

- 台集合は $\mathbb{N}$
- $>$ は自然数上の通常の大小関係
- 任意の $f^{(n)} \in \mathcal{F}$ について $f_{\mathcal{A}}(x_1, \ldots, x_n)$ は n 変数の多項式である。

$(\mathcal{A}, >)$ が多項式解釈のとき $>_{\mathcal{A}}$ を**多項式解釈順序** (polynomial interpretation order) という。

定理 2.25. 多項式解釈順序は簡約順序である。

定義 2.26. 項上の二項関係 R が任意の $C \neq \Box$ なる文脈 C と任意の項 t に対して $C[t]\ R\ t$ を満たすとき、R は**部分項性** (subterm property) を持つという。書換え順序 $>$ が部分項性を持つとき、$>$ は**単純化順序** (simplification order) であるという。

定理 2.27. 有限のシグネチャから成る項上の単純化順序は簡約順序である。

定義 2.28. $>_A$ を集合 A 上の二項関係とする。$>_A$ の**辞書式拡張** (lexicographic extension) $>_A^{\mathsf{lex}}$ は A の要素から成る組上の順序である。$a_1, \ldots, a_m, b_1, \ldots, b_n \in A$ とすると $(a_1, \ldots, a_m) >_A^{\mathsf{lex}} (b_1, \ldots, b_n)$ は以下のいずれかを満たす場合に成り立つ。

- $m > n$
- $m = n > 0$ かつ以下のどちらかが成り立つ。
  1. $a_1 >_A b_1$
  2. $a_1 = b_1$ かつ $(a_2, \ldots, a_m) >_A^{\mathsf{lex}} (b_2, \ldots, b_n)$

定義 2.29. 優先順序 (precedence) $\succ$ をシグネチャ上の狭義半順序とする。**辞書式経路順序** (lexicographic path order) $>_{\mathsf{lpo}}$ は項上の順序として次のように定義される。$s = f(s_1, \ldots, s_n) >_{\mathsf{lpo}} t$ が成り立つとは、以下のいずれかを満たすことである。

- ある $s_i \in \{s_1, \ldots, s_n\}$ が存在して $s_i >_{\mathsf{lpo}}^{=} t$
- $t = g(t_1, \ldots, t_m)$, $f \succ g$ かつすべての $t_j \in \{t_1, \ldots, t_m\}$ に対して $s >_{\mathsf{lpo}} t_j$
- $t = f(t_1, \ldots, t_n)$, $(s_1, \ldots, s_n) >_{\mathsf{lpo}}^{\mathsf{lex}} (t_1, \ldots, t_n)$ かつすべての $t_j \in \{t_1, \ldots, t_m\}$ に対して $s >_{\mathsf{lpo}} t_j$

ただし $>_{\mathsf{lpo}}^{=}$, $>_{\mathsf{lpo}}^{\mathsf{lex}}$ はそれぞれ $>_{\mathsf{lpo}}$ の反射閉包、辞書式拡張である。

定理 2.30. 優先順序が整礎ならば辞書式経路順序は単純化順序である。

定義 2.31. $\mathcal{F}$ をシグネチャとする。**重み関数**（weight function）は関数 $w : \mathcal{F} \to \mathbb{N}$ と、任意の定数記号 $c^{(0)} \in \mathcal{F}$ について $w(c) \geq w_0$ を満たす自然数 w_0 から成る組 (w, w_0) である。項 t の**重み**（weight）$w(t)$ は以下のように定義される。

$$w(t) = \begin{cases} w_0 & t \in \mathcal{V} \text{ のとき} \\ w(f) + \displaystyle\sum_{i=1}^{n} w(t_i) & t = f(t_1, \ldots, t_n) \text{ のとき} \end{cases}$$

定義 2.32. $\succ$ をシグネチャ $\mathcal{F}$ 上の優先順序とする。重み関数 (w, w_0) が $\succ$ に対して**許容的**（admissible）であるとは、次の条件が成り立つことである。$f^{(1)} \in \mathcal{F}$ かつ $w(f) = 0$ を満たすすべての f は、任意の $g \in \mathcal{F} \setminus \{f\}$ に対して $f \succ g$ である。

定義 2.33. $\succ$ を優先順序、(w, w_0) を重み関数とする。**Knuth–Bendix 順序**（Knuth–Bendix order）$>_{\mathsf{kbo}}$ は項上の順序として次のように定義される。$s >_{\mathsf{kbo}} t$ が成り立つとは、すべての $x \in \mathcal{V}$ について $|s|_x \geq |t|_x$ が成り立ち、かつ以下のいずれかを満たすことである。

- $w(s) > w(t)$
- $w(s) = w(t)$ かつ以下のいずれかが成り立つ。
  1. $f^{(1)} \in \mathcal{F}$, $t \in \mathcal{V}$, $n \geq 1$ かつ $s = f^n(t)$
  2. $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$ かつ $(s_1, \ldots, s_n) >_{\mathsf{kbo}}^{\mathsf{lex}} (t_1, \ldots, t_n)$
  3. $s = f(s_1, \ldots, s_m)$, $t = g(t_1, \ldots, t_n)$ かつ $f \succ g$

定理 2.34. 優先順序が整礎かつ重み関数が優先順序に対して許容的であるならば Knuth–Bendix 順序は単純化順序である。

2.5 作用項書換え系

定義 2.35. 作用型シグネチャ（applicative signature）とは、定数と**関数適用記号**（application symbol）と呼ばれる唯一つの二引数関数から構成されるシグネチャである。

本論文では特に言及無く記号 $\circ, \Sigma$ を用いたとき、それぞれ左結合である中置記法の関数適用記号、定数記号の集合を表すことにする。

定義 2.36. 作用型シグネチャ上の項を**作用項**（applicative term）という。作用項から成る項書換え系を**作用項書換え系**（applicative term rewrite system, ATRS）という。

本論文では s, t を作用項としたとき $s \circ t$ という作用項を $\circ$ を省略して $s\ t$ と書くことがある。また、定数記号の集合を単に作用型シグネチャという場合もある。

例 2.37. 以下の $\mathcal{R}$ は作用型シグネチャ $\Sigma = \{\mathsf{id}, \mathsf{add}, \mathsf{s}, 0, \mathsf{map}, :, \mathsf{nil}\}$ 上の作用項書換え系である。

$$\mathcal{R} = \left\{ \begin{array}{c} \mathsf{id}\ x \to x \\ \mathsf{add}\ 0 \to \mathsf{id} \\ \mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) \\ \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{map}\ f\ (:\ x\ \ell) \to\ :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \end{array} \right\}$$

$\circ$ を省略しない場合は以下のように書き表される。

$$\mathcal{R} = \left\{ \begin{array}{c} \mathsf{id} \circ x \to x \\ \mathsf{add} \circ 0 \to \mathsf{id} \\ \mathsf{add} \circ (\mathsf{s} \circ x) \circ y \to \mathsf{s} \circ (\mathsf{add} \circ x \circ y) \\ \mathsf{map} \circ f \circ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{map} \circ f \circ (: \circ x \circ \ell) \to : \circ (f \circ x) \circ (\mathsf{map} \circ f \circ \ell) \end{array} \right\}$$

第 3 章

作用型解釈順序

本章では作用項書換え系のための新たな簡約順序について提案する。

3.1 定義

最初に作用項に関する定義や定理を記述する際に用いる記法を定める。

定義 3.1. f が変数ないし定数であるとき、作用項の組 $(t_1, \ldots, t_n)$ をしばしば $\bar{t}_n$ で表し、$f\ \bar{t}_n$ で $f \circ t_1 \circ \cdots \circ t_n$ を表すものとする。ただし、$n = 0$ の場合 $\bar{t}_n$ は空の組を表し、$f\ \bar{t}_n$ は単に f を表す。作用項 $f\ \bar{t}_n$ に対し、f を**頭部**（head）と呼ぶ

定義 3.2. 2 つの作用項の組 $\bar{s}_m, \bar{t}_n$ に対して $\bar{s}_m \cdot \bar{t}_n$ は $(s_1, \ldots, s_m, t_1, \ldots, t_n)$ という組の連結を表す。また、単一の作用項 t に対して $\bar{s}_m \cdot t$ のように用いられた場合、$(s_1, \ldots, s_m, t)$ を表すものとする。

作用項に対する代数解釈を新たに定義する。この代数解釈では uncurry 化と解釈による比較を同時に行う。uncurry 化は作用項を一階の項へと変換する操作であり、この操作のためには変換後の項書換え系のアリティを定める必要がある。そのため、新たな項書換え系のシグネチャを予め用意するという手法を用いる。

定義 3.3. $\mathcal{F}$ を $\circ$ を含まないシグネチャとする。$\mathcal{F}$ から誘導される**作用型代数**（applicative algebra）は、以下のように定義される新たなシグネチャ $\mathcal{F}^*$ 上の代数である。

$$\mathcal{F}^* = \{f_n^{(n)} \mid f^{(k)} \in \mathcal{F} \text{ かつ } n \in \mathbb{N} \text{ かつ } n \leq k\} \cup \{\circ\}$$

定義 3.4. $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}^*})$ を $\mathcal{F}$ から誘導される作用型代数、α を割り当てとする。

作用型解釈（applicative interpretation）$\langle\alpha\rangle_{\mathcal{A}} : \mathcal{T}(\Sigma \cup \{\circ\}, \mathcal{V}) \to A$ を以下のように帰納的に定める。

$$\langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_n) = \begin{cases} \alpha(f) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t_1) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t_n) & f \in \mathcal{V} \text{ のとき} \\ f_{n\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(t_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(t_n)) & f^{(k)} \in \mathcal{F} \text{ かつ } k \geq n \text{ のとき} \\ \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_{n-1}) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t_n) & f^{(k)} \in \mathcal{F} \text{ かつ } k < n \text{ のとき} \end{cases}$$

作用項 t が基底項である場合、その作用型解釈を割り当てを省略して $\langle t\rangle_{\mathcal{A}}$ と書く場合がある。

例 3.5. シグネチャ $\mathcal{F} = \{\mathsf{add}^{(2)}, \mathsf{s}^{(1)}, \mathsf{id}^{(1)}, 0^{(0)}\}$ に対し、$\mathcal{F}^*$ は以下のようになる。

$$\mathcal{F}^* = \{\mathsf{add}_2, \mathsf{add}_1, \mathsf{add}_0, \mathsf{s}_1, \mathsf{s}_0, \mathsf{id}_1, \mathsf{id}_0, 0_0, \circ\}$$

また、$\mathcal{A}$ を台集合が $\mathbb{N}$ で解釈が以下のように定められている代数とする。

$$\begin{array}{ccc} \mathsf{add}_{2\mathcal{A}}(x,y) = 2x + y + 1, & \mathsf{add}_{1\mathcal{A}}(x) = x + 1, & \mathsf{add}_{0\mathcal{A}} = 1 \\ \mathsf{s}_{1\mathcal{A}}(x) = x + 1, & \mathsf{s}_{0\mathcal{A}} = 1 & \mathsf{id}_{1\mathcal{A}}(x) = x + 1 \\ \mathsf{id}_{0\mathcal{A}} = 1 & 0_{0\mathcal{A}} = 1, & x \circ_{\mathcal{A}} y = 2x + y \end{array}$$

このとき、α を割り当てとし $x, y \in \mathcal{V}$ とすると以下が成り立つ。

$$\begin{array}{lll} \langle\alpha\rangle_{\mathcal{A}}(\mathsf{id}\ x) & = \mathsf{id}_{1\mathcal{A}}(\alpha(x)) & = \alpha(x) + 1 \\ \langle\alpha\rangle_{\mathcal{A}}(\mathsf{add}\ (s\ x)\ y) & = \mathsf{add}_{2\mathcal{A}}(\mathsf{s}_{1\mathcal{A}}(\alpha(x)), \alpha(y)) & = 2\alpha(x) + 2\alpha(y) + 3 \\ \langle\alpha\rangle_{\mathcal{A}}(\mathsf{add}\ x) & = \mathsf{add}_{1\mathcal{A}}(\alpha(x)) & = 2\alpha(x) + 1 \end{array}$$

以下、$\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}^*})$ を $\mathcal{F}$ から誘導される作用型代数、$>$ を A 上の狭義半順序とする。

定義 3.6. 二つの作用項 s, t において、関係 $s \rhd_{\mathcal{A}} t$ が成り立つとは、任意の割り当て α に対して $\langle\alpha\rangle_{\mathcal{A}}(s) > \langle\alpha\rangle_{\mathcal{A}}(t)$ が成り立つことである。

定義 3.7. 二つの作用項 $s = f\ \bar{s}_m$ と $t = g\ \bar{t}_n$ において、関係 $s >_{\mathcal{A}} t$ が成り立つとは、$s \rhd_{\mathcal{A}} t$ が成り立ち、かつ以下の各条件 Var, Sat, Partial のいずれかを満たすことである。

$$\begin{array}{rl} \mathsf{Var} & f \in \mathcal{V}. \\ \mathsf{Sat} & f^{(k)} \in \mathcal{F} \text{ かつ } k \leq m. \\ \mathsf{Partial} & f^{(k)} \in \mathcal{F} \text{ かつ } k > m \text{ かつ } s\ x >_{\mathcal{A}} t\ x. \end{array}$$

ただし、条件 Partial 内の x は s, t に現れない新規変数である。

例 3.8. 作用型シグネチャ $\{\mathsf{add}, \mathsf{id}, \mathsf{s}, 0, \circ\}$ 上の以下の作用項書換え系を考える。

$$\mathsf{id}\ x \to x \tag{3.1}$$
$$\mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) \tag{3.2}$$
$$\mathsf{add}\ 0 \to \mathsf{id} \tag{3.3}$$

例 3.5 の作用型代数 $\mathcal{A}$ を用いて各書換え規則を $>_{\mathcal{A}}$ で向き付ける。α を任意の割り当てとすると以下の不等式が成り立つ。

$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{id}\ x) = \alpha(x) + 1 > \alpha(x) = \langle\alpha\rangle_{\mathcal{A}}(x)$$
$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{add}\ (\mathsf{s}\ x)\ y) = 2\alpha(x) + 2\alpha(y) + 3 > 2\alpha(x) + \alpha(y) + 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{s}\ (\mathsf{add}\ x\ y))$$
$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{add}\ 0) = 1 > 0 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{id})$$
$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{add}\ 0\ x) = \alpha(x) + 3 > \alpha(x) + 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{id}\ x)$$

従って以下の関係が成り立つ。

$$\mathsf{id}\ x \rhd_{\mathcal{A}} x \tag{3.4}$$
$$\mathsf{add}\ (\mathsf{s}\ x)\ y \rhd_{\mathcal{A}} \mathsf{s}\ (\mathsf{add}\ x\ y) \tag{3.5}$$
$$\mathsf{add}\ 0 \rhd_{\mathcal{A}} \mathsf{id} \tag{3.6}$$
$$\mathsf{add}\ 0\ x \rhd_{\mathcal{A}} \mathsf{id}\ x \tag{3.7}$$

$\mathsf{id}^{(1)}, \mathsf{s}^{(1)}, \mathsf{add}^{(2)} \in \mathcal{F}$ かつ (3.4),(3.5),(3.7) であることから条件 Sat より $\mathsf{id}\ x >_{\mathcal{A}} x$, $\mathsf{add}\ (\mathsf{s}\ x)\ y >_{\mathcal{A}} \mathsf{s}\ (\mathsf{add}\ x\ y)$, $\mathsf{add}\ 0\ x >_{\mathcal{A}} \mathsf{id}\ x$ がそれぞれ成り立つ。また、(3.6) かつ $\mathsf{add}\ 0\ x >_{\mathcal{A}} \mathsf{id}\ x$ かつ x は $\mathsf{add}\ 0$, id に現れない変数なので条件 Partial より $\mathsf{add}\ 0 >_{\mathcal{A}} \mathsf{id}$ が成り立つ。

定義 3.7 によれば、$s \rhd_{\mathcal{A}} t$ は $s >_{\mathcal{A}} t$ の必要条件である。更に、条件 Partial の場合は両辺に変数 x を適用した $s\ x >_{\mathcal{A}} t\ x$ が要求される。これは $s\ x \rhd_{\mathcal{A}} t\ x$ も要求されることを意味する。一見これは冗長な比較に見えるが、Partial における $s \rhd_{\mathcal{A}} t$ の条件を省いた場合、$>_{\mathcal{A}}$ の推移性が成り立たない。この事実を以下の例で示す。

例 3.9. 定義 3.7 の条件 Partial の場合は $\rhd_{\mathcal{A}}$ による比較が必要ないと仮定し、作用型シグネチャ $\{\mathsf{f}, \mathsf{g}, \mathsf{h}, \mathsf{a}, \circ\}$ 上の作用項について考える。シグネチャ $\mathcal{F} = \{\mathsf{f}^{(1)}, \mathsf{g}^{(1)}, \mathsf{h}^{(1)}, \mathsf{a}^{(0)}\}$ から誘導される作用型代数 $\mathcal{A}$ の台集合を $\mathbb{N}$, 解釈を以下のように定める。

$$\mathsf{f}_{1\mathcal{A}}(x) = x + 2, \qquad \mathsf{a}_{0\mathcal{A}} = 0$$
$$\mathsf{g}_{1\mathcal{A}}(x) = x + 2, \qquad \mathsf{g}_{0\mathcal{A}} = 1$$
$$\mathsf{h}_{1\mathcal{A}}(x) = x + 1, \qquad \mathsf{h}_{0\mathcal{A}} = 2$$

このとき、次式と $\mathsf{f}^{(1)} \in \mathcal{F}$ と条件 Sat から $\mathsf{f\ a} >_{\mathcal{A}} \mathsf{g}$ が成り立つ。

$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{f\ a}) = 2 > 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{g})$$

また、次に示す通り x を変数として $\mathsf{g}\ x >_{\mathcal{A}} \mathsf{h}\ x$ が成り立ち、かつ $\mathsf{g}^{(1)} \in \mathcal{F}$ と条件 Partial から $\mathsf{g} >_{\mathcal{A}} \mathsf{h}$ が成り立つ。

$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{g}\ x) = \alpha(x) + 2 > \alpha(x) + 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{h}\ x)$$

しかし、下記の通り $\mathsf{f\ a} >_{\mathcal{A}} \mathsf{h}$ は成り立たない。

$$\langle\alpha\rangle_{\mathcal{A}}(\mathsf{f\ a}) = 2 \not> 2 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{h})$$

以下で定義される作用型代数の性質は、$>_{\mathcal{A}}$ が単調性を持つために必要となる。

定義 3.10. 作用型代数 $(\mathcal{A}, >)$ が次の性質を満たしているとき、**関数適用について互換性を持つ**（application compatible）という。すべての $f_n \in \mathcal{F}^*$ について、もし $f_{n+1} \in \mathcal{F}^*$ が存在するならば、以下の不等式が成り立つ。

$$f_{n\mathcal{A}}(a_1, \ldots, a_n) \circ_{\mathcal{A}} b \geq f_{n+1\mathcal{A}}(a_1, \ldots, a_n, b)$$

ただし、$\geq$ は $>$ の反射閉包である。

例 3.11. 例 3.5 の $\mathcal{A}$ は関数適用について互換性を持つ代数である。実際、$x, y \in A$ として以下の不等式群が成り立つ。

$$\mathsf{add}_{0\mathcal{A}} \circ_{\mathcal{A}} x = x + 2 \geq x + 1 = \mathsf{add}_{1\mathcal{A}}(x)$$
$$\mathsf{add}_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y = 2x + y + 2 \geq 2x + y + 1 = \mathsf{add}_{2\mathcal{A}}(x, y)$$
$$\mathsf{s}_{0\mathcal{A}} \circ_{\mathcal{A}} x = x + 2 \geq x + 1 = \mathsf{s}_{1\mathcal{A}}(x)$$
$$\mathsf{id}_{0\mathcal{A}} \circ_{\mathcal{A}} x = x + 2 \geq x + 1 = \mathsf{id}_{1\mathcal{A}}(x)$$

3.2 性質

$>_{\mathcal{A}}$ が文脈について閉じた整礎順序を構成することを示す。以下、$(\mathcal{A}, >)$ を関数適用について互換性を持つ整礎単調代数とする。

補題 3.12. $\rhd_{\mathcal{A}}$ は非反射的である。

証明. 背理法で示す。$\rhd_{\mathcal{A}}$ が非反射的でないと仮定する。このとき、ある作用項 t は $t \rhd_{\mathcal{A}} t$ を満たす。すると、$\rhd_{\mathcal{A}}$ の定義より任意の割り当て α に対して $\langle\alpha\rangle_{\mathcal{A}}(t) > \langle\alpha\rangle_{\mathcal{A}}(t)$ であるが、これは $>$ が非反射的であることに矛盾する。 □

補題 3.13. $>_{\mathcal{A}}$ は非反射的である。

証明. 背理法で示す。$>_{\mathcal{A}}$ が非反射的でないと仮定する。このとき、ある作用項 t は $t >_{\mathcal{A}} t$ を満たす。すると、$>_{\mathcal{A}}$ の定義より $t \rhd_{\mathcal{A}} t$ であるが、これは $\rhd_{\mathcal{A}}$ が非反射的であることに矛盾する。 □

次に、$>_{\mathcal{A}}$ は新規変数のみに影響する代入については閉じていることを示す。

補題 3.14. 任意の $f, g \in \mathcal{V} \cup \Sigma$, 任意の作用項 $s_1, \ldots, s_m, t_1, \ldots, t_n, u$, 新規変数 x に対して、$f\ s_1 \cdots x \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n$ ならば $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ が成り立つ。

証明. y を新規変数として、$f\ s_1 \cdots y \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots y \cdots t_n$ であると仮定する。このとき α を任意の割り当てとすると $\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots y \cdots s_m) > \langle\alpha\rangle_{\mathcal{A}}(g\ t_1 \cdots y \cdots t_n)$ が成り立つ。また、β を以下を満たす割り当てとして定義する。

$$\beta(x) = \begin{cases} \langle\alpha\rangle_{\mathcal{A}}(u) & x = y \text{ のとき} \\ \alpha(x) & \text{それ以外のとき} \end{cases}$$

命題を f の場合分けにより示す。

- $f^{(k)} \in \mathcal{F}$ かつ $k \geq m+1$ の場合、$\langle\beta\rangle_{\mathcal{A}}(f\ s_1 \cdots y \cdots s_m) = \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots u \cdots s_m)$ が以下の通り成り立つ。

$$\begin{aligned} \langle\beta\rangle_{\mathcal{A}}(f\ s_1 \cdots y \cdots s_m) &= f_{k\mathcal{A}}(\langle\beta\rangle_{\mathcal{A}}(s_1), \ldots, \beta(y), \ldots, \langle\beta\rangle_{\mathcal{A}}(s_m)) \\ &= f_{k\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(s_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(u), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_m)) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots u \cdots s_m) \end{aligned}$$

  ただし、上式の 2 つ目の等号は $s_1, \ldots, s_m$ には y が含まれないことから全ての $\varsigma \in \{s_1, \ldots, s_m\}$ なる作用項 ς において $\langle\alpha\rangle_{\mathcal{A}}(\varsigma) = \langle\beta\rangle_{\mathcal{A}}(\varsigma)$ が成り立つことによる。同様に、$\langle\beta\rangle_{\mathcal{A}}(g\ t_1 \cdots y \cdots t_n) = \langle\alpha\rangle_{\mathcal{A}}(g\ t_1 \cdots u \cdots t_n)$ も成り立つ。仮定より、割り当て β についても $\langle\beta\rangle_{\mathcal{A}}(f\ s_1 \cdots y \cdots s_m) > \langle\beta\rangle_{\mathcal{A}}(g\ t_1 \cdots y \cdots t_n)$ が成り立つので $\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots u \cdots s_m) > \langle\alpha\rangle_{\mathcal{A}}(g\ t_1 \cdots u \cdots t_n)$ が成り立つ。従って $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ が成り立つ。
- $k < m+1$ の場合と $f \in \mathcal{V}$ の場合も同様にして $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ であることを示せる。 □

補題 3.15. 任意の $f, g \in \mathcal{V} \cup \Sigma$, 任意の作用項 $s_1, \ldots, s_m, t_1, \ldots, t_n, u$, 新規変数 x に対

して、$f\ s_1 \cdots x \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n$ ならば $f\ s_1 \cdots u \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ が成り立つ。

証明. $f\ s_1 \cdots x \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n$ を仮定する。仮定の必要条件として $f\ s_1 \cdots x \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n$ が成り立つので、u を任意の項とすると補題 3.14 より $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ が成り立つ。従って $f \in \mathcal{V}$ または $f^{(k)} \in \mathcal{F}$ かつ $k < m+1$ の場合は $f\ s_1 \cdots u \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ が成り立つ。残りの $k \geq m+1$ の場合について $N = k - (m+1) \geq 0$ に関する帰納法で示す。

- $N = 0$ のとき、$k = m+1$ かつ $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ より $f\ s_1 \cdots u \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ は成り立つ。
- $N \geq 1$ のとき、$k > m+1$ なので、仮定の不等式 $f\ s_1 \cdots x \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n$ の必要条件として $f\ s_1 \cdots x \cdots s_m\ x' >_{\mathcal{A}} g\ t_1 \cdots x \cdots t_n\ x'$ が成り立つ。ただし x' は新規変数である。よって $f\ s_1 \cdots u \cdots s_m\ x' >_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n\ x'$ が帰納法の仮定より成り立つ。加えて $f\ s_1 \cdots u \cdots s_m \rhd_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ も成り立つので $f\ s_1 \cdots u \cdots s_m >_{\mathcal{A}} g\ t_1 \cdots u \cdots t_n$ は成り立つ。 □

次に、$>_{\mathcal{A}}$ が引数の結合についての単調性（$s >_{\mathcal{A}} t$ ならば $s\ u >_{\mathcal{A}} t\ u$）を持つことを示す。

補題 3.16. $t = f\ \bar{t}_n$ を作用項とする。$f \in \mathcal{V}$ または $f^{(k)} \in \mathcal{F}$ かつ $k \leq n$ のとき、任意の作用項 u について $\langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) = \langle\alpha\rangle_{\mathcal{A}}(t\ u)$ が成り立つ。

証明. $f \in \mathcal{V}$ の場合と $f^{(k)} \in \mathcal{F}$ かつ $k \leq n$ の場合の場合分けで示す。

- $f \in \mathcal{V}$ の場合、以下の通り $\langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) = \langle\alpha\rangle_{\mathcal{A}}(t\ u)$ が成り立つ。

$$\begin{aligned}\langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_n) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) \\ &= \alpha(f) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t_1) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t_n) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_n \cdot u) \\ &= \langle\alpha\rangle_{\mathcal{A}}(t\ u)\end{aligned}$$

- $f^{(k)} \in \mathcal{F}$ かつ $k \leq n$ の場合、$k < n+1$ と $\langle\alpha\rangle_{\mathcal{A}}$ の定義より以下の等式が成り立つ。

$$\begin{aligned}\langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_n) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{t}_n \cdot u) \\ &= \langle\alpha\rangle_{\mathcal{A}}(t\ u)\end{aligned}$$

よってこの場合も $\langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) = \langle\alpha\rangle_{\mathcal{A}}(t\ u)$ が成り立つ。 □

補題 3.17. s, t を作用項とする。任意の割り当て α に対して以下の不等式が成り立つ。

$$\langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \geq \langle\alpha\rangle_{\mathcal{A}}(s\ t)$$

証明. $s = f\ \bar{s}_m$ とし、f に関する場合分けで示す。

- $f \in \mathcal{V}$ の場合、補題 3.16 より $\langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) = \langle\alpha\rangle_{\mathcal{A}}(s\ t)$ が成り立つ。
- $f^{(k)} \in \mathcal{F}$ かつ $k < m+1$ の場合も補題 3.16 より $\langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) = \langle\alpha\rangle_{\mathcal{A}}(s\ t)$ が成り立つ。
- $f^{(k)} \in \mathcal{F}$ かつ $k \geq m+1$ の場合、代数が関数適用について互換性を持つので以下の通り $\langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \geq \langle\alpha\rangle_{\mathcal{A}}(s\ t)$ が成り立つ。

$$\begin{aligned}
\langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{s}_m) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \\
&= f_{m\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(s_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_m)) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \\
&\geq f_{m+1\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(s_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_m), \langle\alpha\rangle_{\mathcal{A}}(t)) \\
&= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{s}_m \cdot t) \\
&= \langle\alpha\rangle_{\mathcal{A}}(s\ t)
\end{aligned}$$

□

補題 3.18. s, t を作用項とする。$s >_{\mathcal{A}} t$ ならば任意の作用項 u に対して $s\ u >_{\mathcal{A}} t\ u$ が成り立つ。

証明. $s = f\ \bar{s}_m >_{\mathcal{A}} t$ を仮定する。このとき $s \rhd_{\mathcal{A}} t$ が仮定の必要条件として成り立つ。命題を $s >_{\mathcal{A}} t$ が導出された規則による場合分けで示す。

- 条件 Var の場合、以下の不等式が成り立つ。

$$\begin{aligned}
\langle\alpha\rangle_{\mathcal{A}}(s\ u) &= \langle\alpha\rangle_{\mathcal{A}}(s) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) \\
&> \langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(u) \\
&\geq \langle\alpha\rangle_{\mathcal{A}}(t\ u)
\end{aligned}$$

  ただし、一行目の等号は $f \in \mathcal{V}$ と補題 3.16、二行目の不等号は代数の単調性、三行目の不等号は補題 3.17 による。従って条件 Var より $s\ u >_{\mathcal{A}} t\ u$ が成り立つ。
- 条件 Sat の場合は $f^{(k)} \in \mathcal{F}$ かつ $k \leq m$ であり、$f \in \mathcal{V}$ の場合と同様にして $\langle\alpha\rangle_{\mathcal{A}}(s\ u) > \langle\alpha\rangle_{\mathcal{A}}(t\ u)$ が成り立つので $s\ u >_{\mathcal{A}} t\ u$ が成り立つ。

- 条件 Partial の場合、新規変数 x について $s\ x >_{\mathcal{A}} t\ x$ が必要条件として成り立つ。よって $t = g\ \bar{t}_n$ とすると $f\ \bar{s}_m \cdot x >_{\mathcal{A}} g\ \bar{t}_n \cdot x$ であり、補題 3.15 より $f\ \bar{s}_m \cdot u >_{\mathcal{A}} g\ \bar{t}_n \cdot u$ が成り立つ。従って $s\ u >_{\mathcal{A}} t\ u$ が成り立つ。 □

次に、補題 3.13 と合わせて $>_{\mathcal{A}}$ が整礎順序であることを示す。

補題 3.19. $>_{\mathcal{A}}$ は推移的である。

証明. $s >_{\mathcal{A}} t$ かつ $t >_{\mathcal{A}} u$ を仮定する。仮定の必要条件として $s \rhd_{\mathcal{A}} t$ かつ $t \rhd_{\mathcal{A}} u$ が成り立つ。α を任意の割り当てとすると、$\langle\alpha\rangle_{\mathcal{A}}(s) > \langle\alpha\rangle_{\mathcal{A}}(t)$ かつ $\langle\alpha\rangle_{\mathcal{A}}(t) > \langle\alpha\rangle_{\mathcal{A}}(u)$ が成り立つので、$>$ の推移性より $\langle\alpha\rangle_{\mathcal{A}}(s) > \langle\alpha\rangle_{\mathcal{A}}(u)$ を満たす。よって $s \rhd_{\mathcal{A}} u$ が成り立つ。$s = f\ \bar{s}_m$ とすると、$f \in \mathcal{V}$ または $f^{(k)} \in \mathcal{F}$ かつ $k < m$ の場合は $s >_{\mathcal{A}} u$ が成り立つ。残りの $k \geq m$ 場合について $N = k - m \geq 0$ についての帰納法で示す。

- $N = k - m = 0$ の場合、$s = f\ \bar{s}_m \rhd_{\mathcal{A}} u$ が成り立つので $s >_{\mathcal{A}} u$ が成り立つ。
- $N = k - m \geq 1$ の場合、新規変数 x について $f\ \bar{s}_m \cdot x >_{\mathcal{A}} t\ x$ が成り立つ。また、$t >_{\mathcal{A}} u$ と補題 3.18 から、$t\ x >_{\mathcal{A}} u\ x$ も成り立つ。従って帰納法の仮定により $f\ \bar{s}_m \cdot x >_{\mathcal{A}} u\ x$ が成り立つ。$f\ \bar{s}_m \rhd_{\mathcal{A}} u$ も成り立つので、$s = f\ \bar{s}_m >_{\mathcal{A}} u$ が成り立つ。 □

補題 3.20. $>_{\mathcal{A}}$ は整礎である。

証明. 背理法により示す。$>_{\mathcal{A}}$ が整礎ではないと仮定する。背理法の仮定より、$t_1 >_{\mathcal{A}} t_2 >_{\mathcal{A}} \cdots$ なる無限列が存在する。このとき、$>_{\mathcal{A}}$ の定義より $t_1 \rhd_{\mathcal{A}} t_2 \rhd_{\mathcal{A}} t_3 \rhd_{\mathcal{A}} \ldots$ である。ここで α を割り当てとすると、$\langle\alpha\rangle_{\mathcal{A}}(t_1) > \langle\alpha\rangle_{\mathcal{A}}(t_2) > \ldots$ なる無限列が存在するが、これは $>$ が整礎であることに矛盾する。 □

次に、$>_{\mathcal{A}}$ が関数の結合についての単調性（$s >_{\mathcal{A}} t$ ならば $u\ s >_{\mathcal{A}} u\ t$）を持つことを示す。

補題 3.21. 任意の $f \in \mathcal{V} \cup \Sigma$ と任意の作用項 $s_1, \ldots, s_n, t$ に対し、$s_i \rhd_{\mathcal{A}} t$ ならば $f\ s_1 \cdots s_i \cdots s_n \rhd_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ が成り立つ。

証明. $s_i \rhd_{\mathcal{A}} t$ を仮定する。このとき、α を任意の割り当てとすると、$\langle\alpha\rangle_{\mathcal{A}}(s_i) > \langle\alpha\rangle_{\mathcal{A}}(t)$ が成り立つ。このとき $\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots s_i \cdots s_n) > \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots t \cdots s_n)$ が成り立つことを f に関する場合分けで示す。

- $f \in \mathcal{V}$ の場合、代数の単調性より以下の不等式が成り立つ。

$$\begin{aligned}\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots s_i \cdots s_n) &= \alpha(f) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_i) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &> \alpha(f) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots t \cdots s_n)\end{aligned}$$

- $f^{(k)} \in \mathcal{F}$ かつ $k < i$ の場合、代数の単調性より以下の不等式が成り立つ。

$$\begin{aligned}\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots s_i \cdots s_n) &= \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{s}_{i-1}) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_i) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &> \langle\alpha\rangle_{\mathcal{A}}(f\ \bar{s}_{i-1}) \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(t) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots t \cdots s_n)\end{aligned}$$

- $f^{(k)} \in \mathcal{F}$ かつ $k \geq i$ の場合、代数の単調性より以下の不等式が成り立つ。

$$\begin{aligned}&\langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots s_i \cdots s_n) \\ &= f_{m\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(s_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_i), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_m)) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &> f_{m\mathcal{A}}(\langle\alpha\rangle_{\mathcal{A}}(s_1), \ldots, \langle\alpha\rangle_{\mathcal{A}}(t), \ldots, \langle\alpha\rangle_{\mathcal{A}}(s_m)) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle\alpha\rangle_{\mathcal{A}}(s_n) \\ &= \langle\alpha\rangle_{\mathcal{A}}(f\ s_1 \cdots t \cdots s_n)\end{aligned}$$

  ただし、m は $i \leq m \leq n$ を満たす自然数である。 □

補題 3.22. 任意の $f \in \mathcal{V} \cup \Sigma$ と任意の作用項 $s_1, \ldots, s_n, t$ に対し、$s_i >_{\mathcal{A}} t$ ならば $f\ s_1 \cdots s_i \cdots s_n >_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ が成り立つ。

証明. $s_i >_{\mathcal{A}} t$ を仮定する。仮定の必要条件として $s_i \rhd_{\mathcal{A}} t$ が成り立つので、補題 3.21 より $f\ s_1 \cdots s_i \cdots s_n \rhd_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ が成り立つ。従って $f \in \mathcal{V}$ または $f^{(k)} \in \mathcal{F}$ かつ $k < n$ の場合は $f\ s_1 \cdots s_i \cdots s_n >_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ が成り立つ。残りの $k \geq n$ の場合について $N = k - n \geq 0$ に関する帰納法で示す。

- $N = 0$ の場合、$k = n$ かつ $f\ s_1 \cdots s_i \cdots s_n \rhd_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ より $f\ s_1 \cdots s_i \cdots s_n >_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ は成り立つ。
- $N \geq 1$ の場合、x を新規変数とすると帰納法の仮定より

$$f\ s_1 \cdots s_i \cdots s_n\ x >_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n\ x$$

  が成り立つ。更に $f\ s_1 \cdots s_i \cdots s_n \rhd_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ も成り立つので、$f\ s_1 \cdots s_i \cdots s_n >_{\mathcal{A}} f\ s_1 \cdots t \cdots s_n$ は成り立つ。 □

補題 3.23. s, t を作用項とする。$s >_{\mathcal{A}} t$ ならば任意の作用項 u に対して $u\ s >_{\mathcal{A}} u\ t$ が成り立つ。

証明. $s >_{\mathcal{A}} t$ を仮定する。$u = f\ \bar{u}_n$ とすると、補題 3.22 より以下の関係が成り立つ。

$$f\ u_1 \cdots u_n\ s >_{\mathcal{A}} f\ u_1 \cdots u_n\ t$$

従って $u\ s >_{\mathcal{A}} u\ t$ は成り立つ。 □

最後に、二つの単調性から $>_{\mathcal{A}}$ が文脈について閉じていることを示す。

補題 3.24. $>_{\mathcal{A}}$ は文脈について閉じている。

証明. $s >_{\mathcal{A}} t$ を仮定し、任意の文脈 C に対して $C[s] >_{\mathcal{A}} C[t]$ が成り立つことを C に関する帰納法で示す。

- $C = \Box$ の場合、$C[s] = s >_{\mathcal{A}} t = C[t]$ より成り立つ。
- $C \neq \Box$ の場合、u を作用項、C' を文脈として $C = C'\ u$ と $C = u\ C'$ の二通りの場合がある。$C = C'\ u$ の場合、帰納法の仮定より $C'[s] >_{\mathcal{A}} C'[t]$ が成り立つことと補題 3.18 より以下の不等式が成り立つ。

  $$C[s] = C'[s]\ u >_{\mathcal{A}} C'[t]\ u = C[t]$$

  $C = u\ C'$ の場合は帰納法の仮定と補題 3.23 より以下の等式が成り立つ。

  $$C[s] = u\ C'[s] >_{\mathcal{A}} u\ C'[t] = C[t]$$

  が成り立つ。 □

3.3 簡約順序

作用型解釈順序 $>_{\mathcal{A}}$ は前節で示したように整礎で文脈について閉じている狭義半順序であるが、代入について閉じていない。このことを例で示す。

例 3.25. シグネチャ $\mathcal{F} = \{\mathsf{f}^{(1)}, \mathsf{a}^{(0)}, \mathsf{b}^{(0)}\}$ から誘導される作用型代数 $\mathcal{A}$ の台集合が $\mathbb{N}$ で以下の解釈を持つとする。

$$\mathsf{f}_{1\mathcal{A}}(x) = x, \qquad \mathsf{a}_{0\mathcal{A}} = 0$$
$$\mathsf{b}_{0\mathcal{A}} = 1 \qquad \circ_{\mathcal{A}}(x, y) = x + y + 2$$

このとき、x を変数とすると、二つの作用型項 $x\ \mathsf{a}$ と b について以下の不等式が成り立っている。

$$\langle\alpha\rangle_{\mathcal{A}}(x\ \mathsf{a}) = \alpha(x) + 2 > 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{b})$$

ただし α は任意の割り当てとする。従って $x\ \mathsf{a} \rhd_{\mathcal{A}} \mathsf{b}$, 更に $x\ \mathsf{a} >_{\mathcal{A}} \mathsf{b}$ が成り立つ。一方、σ を $x\sigma = \mathsf{f}$ を満たす代入とすると、二項 $(x\ \mathsf{a})\sigma$, $\mathsf{b}\sigma$ の間には以下の関係が成り立つ。

$$\langle\alpha\rangle_{\mathcal{A}}((x\ \mathsf{a})\sigma) = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{f}\ \mathsf{a}) = 0 < 1 = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{b}) = \langle\alpha\rangle_{\mathcal{A}}(\mathsf{b}\sigma)$$

従って $(x\ \mathsf{a})\sigma >_{\mathcal{A}} \mathsf{b}\sigma$ は成り立たない。

例 3.25 のような左辺に頭部が変数であるような作用項が出現する場合に対処するために、基底代入を導入した新たな順序を定義する。

定義 3.26. 二つの作用項 s, t について関係 $s \sqsupset_{\mathcal{A}} t$ が成り立つとは、任意の $\tau : \mathcal{V} \to \mathcal{T}(\Sigma \cup \{\circ\})$ なる基底代入 τ に対して $s\tau >_{\mathcal{A}} t\tau$ が満たされていることである。

以下、$\sqsupset_{\mathcal{A}}$ が簡約順序であることを示す。

補題 3.27. $\sqsupset_{\mathcal{A}}$ は代入について閉じている。

証明. $s \sqsupset_{\mathcal{A}} t$ を仮定する。このとき、任意の代入 σ について $s\sigma \sqsupset_{\mathcal{A}} t\sigma$ が成り立つことを示す。τ を任意の基底代入とすると、σ と τ の合成 $\sigma\tau$ は基底代入となる。よって $\sqsupset_{\mathcal{A}}$ の定義により $s(\sigma\tau) >_{\mathcal{A}} t(\sigma\tau)$, つまり $(s\sigma)\tau >_{\mathcal{A}} (t\sigma)\tau$ が成り立つ。従って、$s\sigma \sqsupset_{\mathcal{A}} t\sigma$ が成り立つ。 □

定理 3.28. $\sqsupset_{\mathcal{A}}$ は簡約順序である。

証明. $\sqsupset_{\mathcal{A}}$ が整礎で文脈について閉じている狭義半順序であることは $>_{\mathcal{A}}$ が整礎で文脈について閉じている狭義半順序であることから従う。更に補題 3.27 により $\sqsupset_{\mathcal{A}}$ は代入について閉じている。従って、$\sqsupset_{\mathcal{A}}$ は簡約順序である。 □

3.4 有向性基準

3.3 節で得られた簡約順序 $\sqsupset_{\mathcal{A}}$ は定義に基底代入に関する全称量化子を含み自動化が難しいため、全称量化子が消去された停止性基準を提案する。

定義 3.29. 作用項 $f\ \bar{t}_n$ において、頭部 f が変数であり、かつ $\bar{t}_n$ が一つ以上の項から成る組の場合、f を**頭部変数**（head variable）といい、$f\ \bar{t}_n$ を**頭部変数項**と呼ぶ。頭部変数が出現しない作用項を**無頭部変数項**（head variable free term）という。作用項書換え系 $\mathcal{R}$ の全ての規則の左辺に頭部変数が出現しないとき、$\mathcal{R}$ は**左辺無頭部変数**（left head variable free）であるという。

以下の定義はこの節の証明に用いる。

定義 3.30. $\mathcal{A}$ を台集合が A である作用型代数、τ を基底代入とする。代入 $\tau_{\mathcal{A}} : \mathcal{V} \to A$ を以下のように定める。

$$\tau_{\mathcal{A}}(x) = \langle x\tau \rangle_{\mathcal{A}}$$

以下、$(\mathcal{A}, >)$ を関数適用について互換性を持つ整礎単調代数とする。

補題 3.31. 作用項 t に頭部変数が出現しないとき、以下の等式が成り立つ。

$$\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t) = \langle t\tau \rangle_{\mathcal{A}}$$

証明. t に関する構造帰納法で示す。

- $t \in \mathcal{V}$ の場合、付値関数と $\tau_{\mathcal{A}}$ の定義より、$\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t) = \tau_{\mathcal{A}}(t) = \langle t\tau \rangle_{\mathcal{A}}$ が成り立つ。
- $t = f \in \Sigma$ の場合、付値関数の定義より、$\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t) = f_{0\mathcal{A}} = \langle t\tau \rangle_{\mathcal{A}}$ が成り立つ。
- $t = f\ \bar{t}_n$ の場合、t は無頭部変数項なので $f \in \Sigma$ である。$f^{(k)} \in \mathcal{F}$ とし、$k \geq n$ の場合を示す。帰納法の仮定より $1 \leq i \leq n$ なる全ての i について $\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t_i) = \langle t_i\tau \rangle_{\mathcal{A}}$ なので以下の等式が成り立つ。

$$\begin{aligned}
\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(f\ \bar{t}_n) &= f_{n\mathcal{A}}(\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t_1), \ldots, \langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t_n)) \\
&= f_{n\mathcal{A}}(\langle t_1\tau \rangle_{\mathcal{A}}, \ldots, \langle t_n\tau \rangle_{\mathcal{A}}) \\
&= \langle f\ \bar{t}_n\tau \rangle_{\mathcal{A}} \\
&= \langle (f\ \bar{t}_n)\tau \rangle_{\mathcal{A}}
\end{aligned}$$

よってこの場合は $\langle \tau_{\mathcal{A}} \rangle_{\mathcal{A}}(t) = \langle t\tau \rangle_{\mathcal{A}}$ が成り立つ。$k < n$ の場合も同様にして示せる。 □

補題 3.32. $f\ \bar{s}_m, t_1, \ldots, t_n$ が基底作用項のとき、以下の不等式が成り立つ。

$$\langle f\ \bar{s}_m \rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle t_1 \rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n \rangle_{\mathcal{A}} \geq \langle f\ \bar{s}_m \cdot \bar{t}_n \rangle_{\mathcal{A}}$$

証明. $f\ \bar{s}_m$ は基底項なので $f \in \Sigma$ である。$f^{(k)} \in \mathcal{F}$ とし、k に関する場合分けにより示す。

- $k \leq m$ の場合、$\langle f\ \bar{s}_m \rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle t_1 \rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n \rangle_{\mathcal{A}} = \langle f\ \bar{s}_m \cdot \bar{t}_n \rangle_{\mathcal{A}}$ が成り立つ。
- $k > m$ の場合、代数が関数適用について互換性を持つことから以下の不等式が得

られる。

$$
\begin{aligned}
&\langle f\ \bar{s}_m\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle t_1\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n\rangle_{\mathcal{A}} \\
&= f_{m\mathcal{A}}(\langle s_1\rangle_{\mathcal{A}}, \ldots, \langle s_m\rangle_{\mathcal{A}}) \circ_{\mathcal{A}} \langle t_1\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n\rangle_{\mathcal{A}} \\
&\geq f_{m+1\mathcal{A}}(\langle s_1\rangle_{\mathcal{A}}, \ldots, \langle s_m\rangle_{\mathcal{A}}, \langle t_1\rangle_{\mathcal{A}}) \circ_{\mathcal{A}} \langle t_2\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n\rangle_{\mathcal{A}} \\
&\ldots \\
&\geq f_{k\mathcal{A}}(\langle s_1\rangle_{\mathcal{A}}, \ldots, \langle s_m\rangle_{\mathcal{A}}, \langle t_1\rangle_{\mathcal{A}}, \ldots, \langle t_{k-m}\rangle_{\mathcal{A}}) \circ_{\mathcal{A}} \langle t_{k-m+1}\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle t_n\rangle_{\mathcal{A}} \\
&= \langle f\ \bar{s}_m \cdot \bar{t}_{k-m}\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle t_{k-m+1}\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle t_n\rangle_{\mathcal{A}} \\
&= \langle f\ \bar{s}_m \cdot \bar{t}_n\rangle_{\mathcal{A}}
\end{aligned}
$$

□

以下、左辺の項が無頭部変数項であるとき $>_{\mathcal{A}}$ で $\sqsupset_{\mathcal{A}}$ を模倣できることを示す。

補題 3.33. 任意の作用項 t と基底代入 τ について、以下の不等式が成り立つ。

$$\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t) \geq \langle t\tau\rangle_{\mathcal{A}}$$

証明. t に関する構造帰納法により示す。$t = f\ \bar{t}_n$ とする。

- $n = 0$ の場合、補題 3.31 より $\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t) = \langle t\tau\rangle_{\mathcal{A}}$ が成り立つ。
- $n > 1$ かつ $f \in \Sigma$ の場合、$f^{(k)} \in \mathcal{F}$ であるとすると、帰納法の仮定と代数の単調性より以下の不等式が得られる。

  $$
  \begin{aligned}
  &\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(f\ \bar{t}_n) \\
  &= f_{k\mathcal{A}}(\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_1), \ldots, \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_k)) \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_{k+1}) \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_n) \\
  &= \geq f_{k\mathcal{A}}(\langle t_1\tau\rangle_{\mathcal{A}}, \ldots, \langle t_k\tau\rangle_{\mathcal{A}}) \circ_{\mathcal{A}} \langle t_{k+1}\tau\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \ldots \circ_{\mathcal{A}} \langle t_n\tau\rangle_{\mathcal{A}} \\
  &= \langle (f\ \bar{t}_n)\tau\rangle_{\mathcal{A}}
  \end{aligned}
  $$

  よって、この場合は $\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t) \geq \langle t\tau\rangle_{\mathcal{A}}$ が成り立つ。
- $n > 1$ かつ $f \in \mathcal{V}$ の場合、$f\tau = g\ \bar{s}_m$ とすると、帰納法の仮定と代数の単調性、更に補題 3.32 より以下の不等式が成り立つ。

  $$
  \begin{aligned}
  \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(f\ \bar{t}_n) &= \tau_{\mathcal{A}}(f) \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_1) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_n) \\
  &= \langle g\ \bar{s}_m\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_1) \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t_n) \\
  &\geq \langle g\ \bar{s}_m\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \langle t_1\tau\rangle_{\mathcal{A}} \circ_{\mathcal{A}} \cdots \circ_{\mathcal{A}} \langle t_n\tau\rangle_{\mathcal{A}} \\
  &\geq \langle g\ \bar{s}_m \cdot \bar{t}_n\tau\rangle_{\mathcal{A}} \\
  &= \langle (f\ \bar{t}_n)\tau\rangle_{\mathcal{A}}
  \end{aligned}
  $$

  よって、この場合も $\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t) \geq \langle t\tau\rangle_{\mathcal{A}}$ が成り立つ。

以上より、題意は成立する。 □

定理 3.34. s, t を作用項とする。$s >_{\mathcal{A}} t$ かつ s が無頭部変数項のとき、$s \sqsupset_{\mathcal{A}} t$ が成り立つ。

証明. $s >_{\mathcal{A}} t$ かつ s が無頭部変数項であると仮定する。τ を任意の基底代入とすると、以下の式が成り立つ。

$$\begin{aligned}\langle s\tau\rangle_{\mathcal{A}} &= \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(s)\\ &> \langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}(t)\\ &\geq \langle t\tau\rangle_{\mathcal{A}}\end{aligned}$$

ただし、一行目の等号は s が無頭部変数項であることと補題 3.31、二行目の不等号は仮定 $s >_{\mathcal{A}} t$ と $\langle \tau_{\mathcal{A}}\rangle_{\mathcal{A}}$ が割り当てであること、三行目の不等号は補題 3.33 による。上記の不等式と $s\tau, t\tau$ が基底項であることから $s\tau >_{\mathcal{A}} t\tau$ が成り立つ。従って、$s \sqsupset_{\mathcal{A}} t$ が成り立つ。 □

定理 3.34 より、$>_{\mathcal{A}}$ による停止性基準に関する以下の事実が得られる。

系 3.35. 作用項書換え系 $\mathcal{R}$ が左辺無頭部変数かつ $\mathcal{R} \subseteq >_{\mathcal{A}}$ のとき、$\mathcal{R}$ は停止性を持つ。

証明. 作用項書換え系 $\mathcal{R}$ が左辺無頭部変数かつ $\mathcal{R} \subseteq >_{\mathcal{A}}$ であると仮定する。このとき、定理 3.34 より $\mathcal{R}$ の全ての規則を $\sqsupset_{\mathcal{A}}$ で向き付けることができる。よって $\mathcal{R} \subseteq \sqsupset_{\mathcal{A}}$ であり、$\sqsupset_{\mathcal{A}}$ は簡約順序なので $\mathcal{R}$ は停止性を持つ。 □

第 4 章

自動化

本章では、停止性を証明できる作用型代数の適切な解釈の発見を制約解消系を用いて自動的に行うための手法について述べる。以下の作用項書換え系 $\mathcal{R}_{\mathsf{map}}$ を例として用いる。

$$\mathcal{R}_{\mathsf{map}} = \left\{ \begin{array}{c} \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\ \mathsf{map}\ f\ (:\ x\ \ell) \to :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \end{array} \right\}$$

4.1 代数のクラス

適切な解釈の自動発見のために、用いる代数を多項式解釈に限定する。また、多項式のクラスについても定めておく必要がある。解釈に用いる多項式を線形多項式、つまり $f_{n\mathcal{A}}(x_1, \ldots, x_n) = a_0 + a_1 x_1 + \cdots a_n x_n$ の形式に限定した場合、変数複製への対処が難しく、例えば $\mathsf{times}\ (\mathsf{s}\ x)\ y \to \mathsf{plus}\ (\mathsf{times}\ x\ y)\ y$ のような規則の停止性を証明することができない。よって、各解釈関数 $f_{\mathcal{A}}$ を以下のような多項式と定める。

$$f_{\mathcal{A}}(x_1, \ldots, x_n) = \sum_{I \subseteq \{1,\ldots,n\}} a_I x_I$$

ただし、$x_\varnothing = 1$, $x_{\{i_1,\ldots,i_k\}} = x_{i_1} \cdots x_{i_k}$, 各 I について a_I は自然数係数を表す。このような多項式は、単純混合（simple-mixed）[23] と呼ばれる多項式の部分クラスである。以下の例が示すように、このクラスの多項式解釈により $\mathcal{R}_{\mathsf{map}}$ の停止性を示すことができる。

例 4.1. $\mathcal{F} = \{\mathsf{map}^{(2)}, :^{(2)}, \mathsf{nil}^{(0)}\}$ から誘導される作用型代数 $\mathcal{A}$ を以下の単調な解釈を持

つ多項式解釈とする。

$$\begin{array}{lll} \mathsf{map}_{2\mathcal{A}}(x,y) = xy + x + 3y + 1 & \mathsf{map}_{1\mathcal{A}}(x) = x + 2 & \mathsf{map}_{0\mathcal{A}} = 2 \\ :_{2\mathcal{A}}(x,y) = x + y + 1 & :_{1\mathcal{A}}(x) = x + 1 & :_{0\mathcal{A}} = 1 \\ \mathsf{nil}_{0\mathcal{A}} = 0 & x \circ_{\mathcal{A}} y = xy + x + y + 1 & \end{array}$$

$\mathcal{R}_{\mathsf{map}} \subseteq >_{\mathcal{A}}$ の条件として以下の不等式が成り立つことを確かめる。

$$\begin{array}{c} \mathsf{map}\ f\ \mathsf{nil} >_{\mathcal{A}} \mathsf{nil} \\ \mathsf{map}\ f\ (:\ x\ \ell) >_{\mathcal{A}} :\ (f\ x)\ (\mathsf{map}\ f\ \ell) \end{array}$$

α を任意の割り当てとして以下の不等式が成り立つことを確認すれば良い。

$$\begin{aligned} \langle\alpha\rangle_{\mathcal{A}}(\mathsf{map}\ f\ \mathsf{nil}) &= \mathsf{map}_{2\mathcal{A}}(\alpha(f), \mathsf{nil}_{0\mathcal{A}}) \\ &= 2\alpha(f) + 1 \\ &> 0 \\ &= \mathsf{nil}_{0\mathcal{A}} \\ &= \langle\alpha\rangle_{\mathcal{A}}(\mathsf{nil}) \end{aligned}$$

$$\begin{aligned} \langle\alpha\rangle_{\mathcal{A}}(\mathsf{map}\ f\ (:\ x\ \ell)) &= \mathsf{map}_{2\mathcal{A}}(\alpha(f), :_{2\mathcal{A}}(\alpha(x), \alpha(\ell))) \\ &= \alpha(f)\alpha(x) + \alpha(f)\alpha(\ell) + 2\alpha(f) + 3\alpha(x) + 3\alpha(\ell) + 4 \\ &> \alpha(f)\alpha(x) + \alpha(f)\alpha(\ell) + 2\alpha(f) + \alpha(x) + 3\alpha(\ell) + 3 \\ &= :_{2\mathcal{A}}(\alpha(f) \circ_{\mathcal{A}} \alpha(x), \mathsf{map}_{2\mathcal{A}}(\alpha(f), \alpha(\ell))) \\ &= \langle\alpha\rangle_{\mathcal{A}}(:\ (f\ x)\ (\mathsf{map}\ f\ \ell)) \end{aligned}$$

更に $\mathcal{A}$ が関数適用について互換性を持つことの条件として以下が成り立つことを確かめる。

$$\begin{array}{c} \mathsf{map}_{0\mathcal{A}} \circ_{\mathcal{A}} x \geq \mathsf{map}_{1\mathcal{A}}(x) \\ \mathsf{map}_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y \geq \mathsf{map}_{2\mathcal{A}}(x,y) \\ :_{0\mathcal{A}} \circ_{\mathcal{A}} x \geq :_{1\mathcal{A}}(x) \\ :_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y \geq :_{2\mathcal{A}}(x,y) \end{array}$$

以下の不等式が成り立つことが確認できる。

$$\begin{array}{c} \mathsf{map}_{0\mathcal{A}} \circ_{\mathcal{A}} x = 3x + 3 \geq x + 2 = \mathsf{map}_{1\mathcal{A}}(x) \\ \mathsf{map}_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y = xy + x + 3y + 3 \geq xy + x + 3y + 1 = \mathsf{map}_{2\mathcal{A}}(x,y) \\ :_{0\mathcal{A}} \circ_{\mathcal{A}} x = 2x + 2 \geq x + 1 = :_{1\mathcal{A}}(x) \\ :_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y = xy + x + 2y + 2 \geq x + y + 1 = :_{2\mathcal{A}}(x,y) \end{array}$$

加えて $\mathcal{R}_{\mathsf{map}}$ は左辺無頭部変数なので、系 3.35 より停止性を持つ。

4.2 多項式の非負性判定

前節で行った停止性証明の自動化を考える。与えられた作用項書換え系が左辺無頭部変数かどうかの判定は項の構造を調べることにより容易に行えるため、適切な代数の発見の自動化を主に考える。作用型代数には何らかの方法でシグネチャを指定する必要があるが、最も自然な選び方として [13] で定義されている**作用型アリティ**（applicative arity）を用いる。作用項書換え系 $\mathcal{R}$ において、定数記号 c の作用型アリティ $\mathsf{aa}_{\mathcal{R}}(c)$ は以下のように定義される。

$$\mathsf{aa}_{\mathcal{R}}(c) = \max\{n \mid c\ t_1 \cdots\ t_n \text{ は}\mathcal{R}\text{に含まれる部分項}\}$$

例えば $\mathcal{R}_{\mathsf{map}}$ では、各定数記号の作用型アリティはそれぞれ $\mathsf{aa}_{\mathcal{R}}(\mathsf{map}) = 2$, $\mathsf{aa}_{\mathcal{R}}(:) = 2$, $\mathsf{aa}_{\mathcal{R}}(\mathsf{nil}) = 0$ なので、作用型代数に用いるシグネチャを $\{\mathsf{map}^{(2)}, :^{(2)}, \mathsf{nil}^{(0)}\}$ と定める。

例 4.1 のような解釈の発見の自動化には多項式がしばしば用いられる。$x_1, \ldots, x_n$ を変数に持つ多項式 P, Q は、すべての $a_1, \ldots, a_n \in \mathbb{N}$ に対して $P(a_1, \ldots, a_n) > Q(a_1, \ldots, a_n)$ が成り立つならば $P(x_1, \ldots, x_n) > Q(x_1, \ldots, x_n)$ である。$f, x, \ell \in \mathbb{N}$ として、例 4.1 内の $\mathcal{R}_{\mathsf{map}}$ の各規則の解釈に関する不等式は以下の不等式へ言い換えることができる。

$$2f + 1 > 0$$
$$fx + f\ell + 2f + 3x + 3\ell + 4 > fx + f\ell + 2f + x + 3\ell + 3$$

不等式 $P > Q$ が成り立つには $P - Q - 1$ が非負であればよい。上の不等式において、明らかに $(2f+1)-0-1 = 2f$, $(fx+f\ell+2f+3x+3\ell+4)-(fx+f\ell+2f+x+3\ell+3)-1 = 2x$ は非負である。結局、ある多項式解釈 $\mathcal{A}$ が作用項書換え系 $\mathcal{R}$ に対して $\mathcal{R} \subseteq >_{\mathcal{A}}$ を満たすことを判定する場合、$\mathcal{R}$ の各書換え規則について左辺と右辺の解釈の差が正になることを確かめれば十分である。よって、以下の例が示すように適切な多項式の係数を求める問題を解くことにより、適切な解釈を発見できる。

例 4.2. 未知係数 m_i, c_i, n_0, a_i に対して、解釈を以下のように定める。

$$\mathsf{map}_{2\mathcal{A}}(x, y) = m_0 + m_1 x + m_2 y + m_3 xy \qquad :_{2\mathcal{A}}(x, y) = c_0 + c_1 x + c_2 y + c_3 xy$$
$$\mathsf{nil}_{0\mathcal{A}} = n_0 \qquad \circ_{\mathcal{A}}(x, y) = a_0 + a_1 x + a_2 y + a_3 xy$$
$$\mathsf{map}_{1\mathcal{A}}(x) = m'_0 + m'_1 x \qquad :_{1\mathcal{A}}(x) = c'_0 + c'_1 x$$
$$\mathsf{map}_{0\mathcal{A}} = m''_0 \qquad :_{0\mathcal{A}} = c''_0$$

$\mathcal{R}_{\mathsf{map}}$ の各規則について左辺と右辺の解釈の差から 1 を引いたものを表す式として、不定項 f, x, ℓ に関する以下の多項式が得られる。

$$
\begin{aligned}
& \mathsf{map}_{2\mathcal{A}}(f, \mathsf{nil}_{0\mathcal{A}}) - \mathsf{nil}_{0\mathcal{A}} - 1 \\
= {} & m_0 + m_2 n_0 - n_0 - 1 + (m_1 + m_3 n_0) f \\
& \mathsf{map}_{2\mathcal{A}}(f, :_{2\mathcal{A}}(x, \ell)) - :_{2\mathcal{A}}(f \circ_{\mathcal{A}} x, \mathsf{map}_{2\mathcal{A}}(f, \ell)) - 1 \\
= {} & m_0 + c_0 m_2 - c_0 - a_0 c_1 - c_2 m_0 - a_0 c_3 m_0 - 1 \\
& + (m_1 + c_0 m_3 - a_1 c_1 - c_2 m_1 - a_0 c_3 m_1 - a_1 c_3 m_0) f \\
& + (c_1 m_2 - a_2 c_1 - a_2 c_3 m_0) x \\
& - a_0 c_3 m_2 \ell \\
& + (c_1 m_3 - a_3 c_1 - a_2 c_3 m_1 - a_3 c_3 m_0) f x \\
& + (-a_1 c_3 m_2 - a_0 c_3 m_3) f \ell \\
& + (c_3 m_2 - a_2 c_3 m_2) x \ell \\
& + (c_3 m_3 - a_2 c_3 m_3 - a_3 c_3 m_2) f x \ell \\
& - a_1 c_3 m_1 f^2 - a_3 c_3 m_1 f^2 x - a_1 c_3 m_3 f^2 \ell - a_3 c_3 m_3 f^2 x \ell
\end{aligned}
$$

上記の多項式が非負となるように未知係数の値を定めることにより、適切な解釈が求められる。

一般的に多項式の非負性判定問題は決定不能であるが、いくつかの近似手法が知られている。本研究では Hong と Jakuš [14] が提案した shifting method と呼ばれる手法を用いる。この手法は、多項式のすべての係数の非負性が多項式の非負性の十分条件であるという事実に基づく。例 4.2 で得られた多項式の非負性判定問題は、shifting method を用

いることで不定項に関する全称量化子が消去された以下の制約式に帰着できる。

$$
\begin{aligned}
& m_0 + m_2 n_0 - n_0 - 1 \geq 0 \\
\wedge \quad & m_1 + m_3 n_0 \geq 0 \\
\wedge \quad & m_0 + c_0 m_2 - c_0 - a_0 c_1 - c_2 m_0 - a_0 c_3 m_0 - 1 \geq 0 \\
\wedge \quad & m_1 + c_0 m_3 - a_1 c_1 - c_2 m_1 - a_0 c_3 m_1 - a_1 c_3 m_0 \geq 0 \\
\wedge \quad & c_1 m_2 - a_2 c_1 - a_2 c_3 m_0 \geq 0 \\
\wedge \quad & -a_0 c_3 m_2 \geq 0 \\
\wedge \quad & c_1 m_3 - a_3 c_1 - a_2 c_3 m_1 - a_3 c_3 m_0 \geq 0 \\
\wedge \quad & -a_1 c_3 m_2 - a_0 c_3 m_3 \geq 0 \\
\wedge \quad & c_3 m_2 - a_2 c_3 m_2 \geq 0 \\
\wedge \quad & c_3 m_3 - a_2 c_3 m_3 - a_3 c_3 m_2 \geq 0 \\
\wedge \quad & -a_1 c_3 m_1 \geq 0 \\
\wedge \quad & -a_3 c_3 m_1 \geq 0 \\
\wedge \quad & -a_1 c_3 m_3 \geq 0 \\
\wedge \quad & -a_3 c_3 m_3 \geq 0
\end{aligned}
$$

また、$\mathcal{A}$ が関数適用について互換性を持つための条件式も同様にして不定項がない制約式に帰着できる。例 4.2 で定めた解釈を用いると、以下の不等式が非負であることが条件となる。

$$
\begin{aligned}
& \mathsf{map}_{0\mathcal{A}} \circ_{\mathcal{A}} x - \mathsf{map}_{1\mathcal{A}}(x) \\
= \; & a_0 + a_1 m_0'' - m_0' + (a_2 + a_3 m_0'' - m_1')x \\
& \mathsf{map}_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y - \mathsf{map}_{2\mathcal{A}}(x, y) \\
= \; & a_0 + a_1 m_0' - m_0 + (a_1 m_1' - m_1)x + (a_2 + a_3 m_0' - m_2)y + (a_3 m_1' - m_3)xy \\
& :_{0\mathcal{A}} \circ_{\mathcal{A}} x - :_{1\mathcal{A}}(x) \\
= \; & a_0 + a_1 c_0'' - c_0' + (a_2 + a_3 c_0'' - c_1')x \\
& :_{1\mathcal{A}}(x) \circ_{\mathcal{A}} y - :_{2\mathcal{A}}(x, y) \\
= \; & a_0 + a_1 c_0' - c_0 + (a_1 c_1' - c_1)x + (a_2 + a_3 c_0' - c_2)y + (a_3 c_1' - c_3)xy
\end{aligned}
$$

以下は上記の多項式の非負性判定問題に対応する、不定項を含まない制約式である。

$$
\begin{aligned}
& a_0 + a_1 m_0'' - m_0' \geq 0 \\
\wedge\ & a_2 + a_3 m_0'' - m_1' \geq 0 \\
\wedge\ & a_0 + a_1 m_0' - m_0 \geq 0 \\
\wedge\ & a_1 m_1' - m_1 \geq 0 \\
\wedge\ & a_2 + a_3 m_0' - m_2 \geq 0 \\
\wedge\ & a_3 m_1' - m_3 \geq 0 \\
\wedge\ & a_0 + a_1 c_0'' - c_0' \geq 0 \\
\wedge\ & a_2 + a_3 c_0'' - c_1' \geq 0 \\
\wedge\ & a_0 + a_1 c_0' - c_0 \geq 0 \\
\wedge\ & a_1 c_1' - c_1 \geq 0 \\
\wedge\ & a_2 + a_3 c_0' - c_2 \geq 0 \\
\wedge\ & a_3 c_1' - c_3 \geq 0
\end{aligned}
$$

更に、解釈が単調であるための条件として各解釈の次数 1 の項の係数が正であるという以下の条件も加える。

$$m_1 > 0 \wedge m_2 > 0 \wedge c_1 > 0 \wedge c_2 > 0 \wedge a_1 > 0 \wedge a_2 > 0 \wedge m_1' > 0 \wedge c_1' > 0$$

最終的に得られる制約式は、これまでに得られた制約式を同時に満たす、未知係数に関する充足可能性問題である。この問題は非線形整数算術 (NIA) を理論とする SMT（Satisfiability Modulo Theories）ソルバで解くことができ、実際、次の解を見つけることができる。

$$
\begin{array}{llll}
m_0 = 0, & m_1 = 1, & m_2 = 10, & m_3 = 14 \\
c_0 = 15, & c_1 = 1, & c_2 = 14, & c_3 = 0 \\
a_0 = 6, & a_1 = 2, & a_2 = 1, & a_3 = 14 \\
n_0 = 13 & & & \\
m_0' = 1, & m_1' = 1, & c_0' = 9 & c_1' = 1 \\
m_0'' = 2, & c_0'' = 9 & &
\end{array}
$$

第 5 章

実験と評価

本研究で提案した簡約順序による停止性証明能力を他の手法やツールと比較して評価する。

5.1 実験と評価

3 章で定義した簡約順序に基づく停止性判定ツールを 4 章で述べた手法で実装を行った。実装にはプログラミング言語として Haskell を用いた。多項式解釈の発見に用いる SMT ソルバは Z3 4.8.10 [10] を用いた。問題集として Termination Problem Database (TPDB) [1] version 11.0 の TRS_Standard のうち、作用項書換え系であるものを抽出した。その際、定数記号と 1 つ以下の 2 引数関数記号でシグネチャが構成されているものを作用項書換え系と判断した。問題数は 203 問である。実験に使用した PC は Intel Core i7-1065G7 1.30GHz のプロセッサ及び 16GB のメモリを搭載したものである。計算時間は 1 問につき最大 60 秒とした。以降の表では横軸に停止性証明手法を表す名前を並べ、縦軸には各入力に対して出力したステータスの数を並べている。各ステータス名は入力された TRS に対しての以下の実行結果を表す。

YES	停止性を持つことの証明に成功した。
NO	停止性を持たないことの証明に成功した。
MAYBE	60 秒が経過する前にツールが証明を諦めた。
TIMEOUT	ツールの処理が 60 秒を超えた。

最初に既存の簡約順序との比較を行う。比較に用いた順序を以下に記す。

apol　本研究で提案した作用型解釈順序。

pol　多項式解釈順序。

epo　埋め込み経路順序。5.2 節で詳述する。

lpo　辞書式経路順序。

kbo　Knuth–Bendix 順序。

wpo　重み付き経路順序 [26]。

上記の順序のうち apol, pol, wpo は代数解釈を用いる順序である。4 章 で述べたように apol では多項式のクラスが単純混合である多項式解釈を用いているため、pol と wpo でも同様の解釈を用いている。実験の結果を表 5.1 に記す。apol は他の順序と比べて停止性証明に成功した作用項書換え系の数が増えている。これは他の順序が単純化順序であるため変数複製を伴う作用項書換え系に対処できないが、apol は対処できることに起因すると推測される。問題集の中で変数複製を伴うものは 110 問あり、apol はそのうちの 30 問に対して停止性を証明することができた。

表 5.1　既存の簡約順序との比較

ステータス	apol	pol	epo	lpo	kbo	wpo
YES	**40**	**12**	**6**	**9**	**10**	**12**
total time (sec)	*24.88*	*9.09*	*0.07*	*0.11*	*0.12*	*8.10*
NO	0	0	0	0	0	0
total time (sec)	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*	*0.26*
MAYBE	157	169	194	191	191	150
total time (sec)	*246.59*	*889.87*	*2.68*	*19.38*	*4.99*	*751.99*
TIMEOUT	6	22	3	3	2	41
total time (sec)	*360.00*	*1320.00*	*180.00*	*180.00*	*120.00*	*2460.00*

また、代数を用いる手法である apol, pol, wpo の結果に注目すると、TIMEOUT となった問題数が増えている。これは、apol が uncurry 化によって木構造としての深さが小さくなった項を扱えることに起因すると考えられる。例えば項 $\circ(\circ(:, \circ(f, x)), \circ(\circ(\mathsf{map}, f), \ell))$ の深さは 3 であるのに対して uncurry 化されたもの $:_2(\circ(f, x), \mathsf{map}(f, \ell))$ の深さは 2 となる。4.2 節で見たように、解釈の自動発見のために SMT ソルバに渡す制約式は未知係

数に関する非線形整数算術式であるが、その次数は規則に含まれる項の深さが増すほど指数関数的に大きくなる。実際 4.2 節で扱った map 規則の場合 apol では 3 次の算術式、pol では 6 次の算術式を生成しており、より高次な算術式を渡したために pol と wpo は解の探索に時間がかかってしまったと思われる。

一方、apol は左辺に頭部変数が出現する書換え規則に対しては停止性を証明することができない。問題集にはそのような規則を含む問題が 57 問あり、それらには対処できなかった。

次に、既存の停止性証明ツール TTT2[17], NaTT [25], AProVE [12] と作用型解釈順序 apol の実行結果を比較した。結果を表 5.2 に記す。apol とツールとでは停止性証明能力

表 5.2　停止性証明ツールとの比較

ステータス	apol	TTT2	NaTT	AProVE
YES	**40**	**144**	**146**	**167**
total time (sec)	*24.88*	*33.16*	*7.75*	*318.16*
NO	0	14	9	13
total time (sec)	*0.00*	*11.52*	*3.27*	*59.40*
MAYBE	157	43	46	0
total time (sec)	*205.21*	*246.59*	*13.94*	*0*
TIMEOUT	6	2	2	23
total time (sec)	*360.00*	*120.00*	*120.00*	*1380.00*

に大きな差があることがわかった。これは、ツールは uncurry 化と依存対法 [2] と呼ばれる強力な変換手法を組み合わせて解くことができるためである。本研究で提案した手法は uncurry 化と解釈順序を用いた停止性証明手法を 1 つの順序として再構築したが、更に依存対法をこれに組み込むことができればより広い作用項書換え系のクラスに対して有効な簡約順序になると考えられる。

5.2　関連研究

本研究では高階関数を扱える書換え系の停止性を保証するための新たな順序を提案した。しかし本論文が最初の成果ではない。以下、既存の関連研究について述べる。

■埋め込み経路順序　Bentkamp は作用項上の基底全順序な順序として埋め込み経路順序 [6] を提案した。この順序は superposition calculus [4] と呼ばれる等式推論手法の高階論理版 [7] において、探索空間を制限するために用いる順序として考案された。定義は以下である。

定義 5.1. 2 つの作用項 s, t に対して、埋め込み経路順序 $s >_{\mathsf{ep}} t$ は以下のいずれかを満たすときに成り立つ。

$$\frac{n > 0 \quad chop(s) \geq_{\mathsf{ep}} t}{s = f\ \bar{s}_n >_{\mathsf{ep}} t}$$

$$\frac{f, g \in \Sigma \quad f \succ g \quad n = 0 \vee s >_{\mathsf{ep}} chop(t)}{s >_{\mathsf{ep}} g\ \bar{t}_n = t}$$

$$\frac{f \in \mathcal{V} \quad \bar{s}_m >_{\mathsf{ep}}^{\mathsf{lex}} \bar{t}_n \quad m > 0 \quad n = 0 \vee chop(s) >_{\mathsf{ep}} chop(t)}{s = f\ \bar{s}_m >_{\mathsf{ep}} f\ \bar{t}_n = t}$$

$$\frac{f \in \Sigma \quad \bar{s}_n >_{\mathsf{ep}}^{\mathsf{lex}} \bar{t}_n \quad n = 0 \vee s >_{\mathsf{ep}} chop(t)}{s >_{\mathsf{ep}} g\ \bar{t}_n = t}$$

ただし、$\succ$ は関数記号上の優先順序、$>_{\mathsf{ep}}^{\mathsf{lex}}$ は $>_{\mathsf{ep}}$ の辞書式拡張、$chop$ を以下のように定義される作用項から作用項への部分関数とする。

$$chop(f\ t_1 \cdots t_n) = t_1 \cdots t_n$$

この順序は変数が複製される向きに順序付けることはできないという問題点がある。例えば $(\mathsf{double}\ x, \mathsf{add}\ x\ x)$ という作用項の組を考えると、$\mathsf{add}\ x\ x >_{\mathsf{ep}} \mathsf{double}\ x$ という向き付けは $\mathsf{add} \succ \mathsf{double}$ という優先順序のもとで以下に示す通り行うことができる。

$$\dfrac{\mathsf{add} \succ \mathsf{double} \quad \dfrac{\mathsf{add} \succ \mathsf{double}}{\mathsf{add}\ x\ x >_{\mathsf{ep}} \mathsf{double}}}{\mathsf{add}\ x\ x >_{\mathsf{ep}} \mathsf{double}\ x}$$

しかし、$\mathsf{double}\ x >_{\mathsf{ep}} \mathsf{add}\ x\ x$ という向き付けはどのような優先順序のもとでも不可能である。そのため、$\mathsf{double}\ x \to \mathsf{add}\ x\ x$ という規則の停止性を証明することができない。$\mathsf{map}\ f\ (:\ x\ \ell) \to :\ (f\ x)\ (\mathsf{map}\ f\ \ell)$ のような規則に対しても同様に、望むような順序付けを行うことはできない。作用項書換え系では再帰的な高階関数が多く扱われるが、そのような規則は必ず高階変数の複製を伴うため、埋め込み経路順序は停止性証明に関してはあまり強力ではない。本研究で提案した作用型解釈順序はこの欠点を克服しており、上で挙げた書換え規則の停止性証明を行うことが可能である。

■lambda-free Knuth-Bendix 順序　lambda-free Knuth–Bendix 順序 [5] は前節で紹介した埋め込み経路順序の前身で、一階の Knuth–Bedix 順序 [3] の作用項版である。定義は [5] を参照されたい。一階の Knuth–Bendix 順序と同様、最初に両辺の重みを比較し、重みが等しい場合はルート位置の関数記号を関数記号上の優先順序で比較し、それでも順序が付かない場合は引数を辞書式順序で比較する。一階の項が一階の Knuth–Bendix 順序で向きづけられたとき、その際に用いた重み関数、優先順序を用いることで curry 化された項に対して lambda-free Knuth–Bendix 順序 $>_{\mathsf{kbo}}$ で同じ向き付けを行うことができる。この性質は gracefulness と呼ばれ、一階の superposition calculus [4] で証明できる等式が高階の superposition calculus [7] でも証明できることを保証する。ただし、lambda-free Knuth–Bendix 順序も埋め込み経路順序と同様に、変数複製を伴う向きへの向き付けを行うことができない。そのため $\mathsf{add}\ x\ x >_{\mathsf{kbo}} \mathsf{double}\ x$ は可能であるが、逆向きは不可能である。map も同様である。

■functional　高階関数を扱うための項書換え系の枠組みの一つとして、Nipkow の高階書換え系（higher-order rewrite system, HRS）[20] がある。高階書換えは書換え規則によるパターンマッチングと単純型付きラムダ計算を組み合わせた計算モデルである。高階書換え系に対して、van de Pol は 1996 年に functional [24] と呼ばれる高階関数を解釈とする代数体系を用いて解釈順序に相当する簡約順序を構成する手法提案した。例として以下の高階書換え系 $\mathcal{H}$ について functional による解釈順序をみる。ただし、λ は関数抽象を表す記号、大文字のアルファベットは自由変数、各関数記号の型を $\mathsf{map} : (\mathbf{nat} \to \mathbf{nat}) \to \mathbf{list} \to \mathbf{list}$, $: : \mathbf{nat} \to \mathbf{list} \to \mathbf{list}$, $\mathsf{nil} : \mathbf{list}$ と定める。分野の慣習として $f\ t_1 \cdots t_n$ は $f(t_1, \ldots, t_n)$ と表記される。

$$\mathcal{H} = \left\{ \begin{array}{c} \mathsf{map}((\lambda x.\ F(x)), \mathsf{nil}) \to \mathsf{nil} \\ \mathsf{map}((\lambda x.\ F(x)), :(X, L)) \to :(F(X), \mathsf{map}((\lambda x.\ F(x)), L)) \end{array} \right\}$$

$\mathcal{H}$ の停止性は以下の関数解釈 γ により証明できる。

$$\gamma(\mathsf{map}) = \lambda f \in \mathbb{N} \Rightarrow \mathbb{N}.\ \lambda n \in \mathbb{N}.\ \sum_{i=1}^{n} f(i) + 3n + 1$$

$$\gamma(:) = \lambda m \in \mathbb{N}.\ \lambda n \in \mathbb{N}.\ m + n + 1$$

$$\gamma(\mathsf{nil}) = 1$$

ここで $\mathbb{N} \Rightarrow \mathbb{N}$ は自然数上の単調関数全体の集合を表す。よって $f \in \mathbb{N} \Rightarrow \mathbb{N}$ は f が自然数上の任意の単調関数であることを表している。このとき、$\mathcal{H}$ の各規則は以下のように向

き付けることができる。ただし、$n, l \in \mathbb{N}$, $f \in \mathbb{N} \Rightarrow \mathbb{N}$ とし、$\alpha = \{F \mapsto f, X \mapsto n, L \mapsto \ell\}$ は割り当てとする。項 t の解釈を $[\![\alpha]\!](t)$ で表せば

$$\begin{aligned}
[\![\alpha]\!](\mathsf{map}((x.\ F(x)), \mathsf{nil})) &= \sum_{i=0}^{1} f(i) + 3 + 1 \\
&> 1 \\
&= [\![\alpha]\!](\mathsf{nil}) \\
[\![\alpha]\!](\mathsf{map}((x.\ F(x)), :(X, L))) &= \sum_{i=0}^{3n+3l+3} f(i) + 3n + 3l + 3 + 1 \\
&= \sum_{i=0}^{l} f(i) + \sum_{i=l+1}^{3n+3l+3} f(i) + 3n + 3l + 4 \\
&> f(n) + \sum_{i=0}^{l} f(i) + 3l + 2 \\
&= [\![\alpha]\!](:(F(X), \mathsf{map}((x.\ F(x)), L)))
\end{aligned}$$

上記のように、map に対する関数解釈は通常の多項式とすることができず、停止性証明のための解釈の自動発見が難しい。2012 年に $f(x)$ のような関数適用を不等式から除去する制約式の変換手法 [11] が考案され、4 章で紹介した自動化手法のように非線形算術式の充足可能性問題に帰着する方法が高階書換え系の停止性ツールでは採用されている。

第 6 章

結論

本論文では作用項書換え系のための新たな簡約順序である作用型解釈順序とそれを用いた停止性証明の自動化手法を提案した。この順序は単純化順序ではないため既存手法において対処が難しかった変数複製を伴う方向の向き付けも可能であり、実験によりその有用性を示すことができた。

一方、左辺に頭部変数が現れる規則に対しては今回の手法では扱えていない。しかし、例えば宣言型プログラミング言語において左辺に頭部変数が現れる関数宣言は ill-formed であるように、応用においてこの制限が問題になるケースは少ないと思われる。

作用型解釈順序は uncurry 化と解釈順序を組み合わせることにより作ることができたが、組み合わせる順序によっては更に強力な順序を作ることができると考えられる。例えば、5.2 節で Knuth–Bendix 順序（KBO）の作用項版について述べたが、uncurry 化を埋め込む本アプローチで構成し直すことは検討に値する。もし uncurry 化 KBO を 1 つの順序として再構築できれば、更に解釈順序、KBO, 辞書式経路順序を包含する順序である重み付き経路順序 (WPO) についても同様のアプローチで uncurry 化 WPO への再構成が可能であると考えられる。このことに対する検証は今後の課題である。

■謝辞　研究に際し熱心にご指導して下さった廣川直准教授に心より感謝を申し上げます。中間審査の際に助言を下さった緒方和博教授、石井大輔准教授に感謝を申し上げます。研究活動を支えて下さった研究室の皆様に御礼を申し上げます。

参考文献

[1] The termination problems data base. `http://termination-portal.org/wiki/TPDB`.

[2] Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theor. Comput. Sci.*, 236(1-2):133–178, 2000.

[3] Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.

[4] Leo Bachmair and Harald Ganzinger. On restrictions of ordered paramodulation with simplification. In Mark E. Stickel, editor, *10th International Conference on Automated Deduction, Kaiserslautern, FRG, July 24-27, 1990, Proceedings*, volume 449 of *Lecture Notes in Computer Science*, pages 427–441. Springer, 1990.

[5] Heiko Becker, Jasmin Christian Blanchette, Uwe Waldmann, and Daniel Wand. A transfinite knuth-bendix order for lambda-free higher-order terms. In Leonardo de Moura, editor, *Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings*, volume 10395 of *Lecture Notes in Computer Science*, pages 432–453. Springer, 2017.

[6] Alexander Bentkamp. The embedding path order for lambda-free higher-order terms. *FLAP*, 8(10):2447–2470, 2021.

[7] Alexander Bentkamp, Jasmin Chreistian Blanchette, Simon Cruanes, and Uwe Waldmann. Superposition for lambda-free higher-order logic. *Log. Methods Comput. Sci.*, 17(2), 2021.

[8] Jasmin Christian Blanchette, Sascha Böhme, Andrei Popescu, and Nicholas Smallbone. Encoding monomorphic and polymorphic types. *Log. Methods Comput. Sci.*, 12(4), 2016.

[9] Simon Cruanes. *Extending superposition with integer arithmetic, structural induction, and beyond*. PhD thesis, École polytechnique, 2015.

[10] Leonardo Mendonça de Moura and Nikolaj S. Bjørner. Z3: an efficient SMT solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340. Springer, 2008.

[11] Carsten Fuhs and Cynthia Kop. Polynomial interpretations for higher-order rewriting. In Ashish Tiwari, editor, *23rd International Conference on Rewriting Techniques and Applications (RTA'12) , RTA 2012, May 28 - June 2, 2012, Nagoya, Japan*, volume 15 of *LIPIcs*, pages 176–192. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.

[12] Jürgen Giesl, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Steffi Swiderski, and René Thiemann. Proving termination of programs automatically with aprove. In *IJCAR*. Springer, January 2014.

[13] Nao Hirokawa, Aart Middeldorp, and Harald Zankl. Uncurrying for termination and complexity. *J. Autom. Reason.*, 50(3):279–315, 2013.

[14] Hoon Hong and Dalibor Jakus. Testing positiveness of polynomials. *J. Autom. Reason.*, 21(1):23–38, 1998.

[15] Sam Kamin and Jean-Jacques Lévy. Two generalizations of the recursive path ordering, february 1980. *Unpublished note, Department of Computer Science, University of Illinois, Urbana, IL. Available at http://www. ens-lyon. fr/LIP/REWRITING/OLD PUBLICATIONS ON TERMINATION/KAMIN LEVY (viewed June 2004).*

[16] Donald E Knuth and Peter B Bendix. Simple word problems in universal algebras. computational problems in abstract algebra. *Leech, J.(ed.)*, pages 263–297, 1970.

[17] Martin Korp, Christian Sternagel, Harald Zankl, and Aart Middeldorp. Tyrolean termination tool 2. pages 295–304, 06 2009.

[18] Laura Kovács and Andrei Voronkov. First-order theorem proving and vampire. In *International Conference on Computer Aided Verification*, pages 1–35. Springer,

2013.

[19] Dallas Lankford. On proving term rewriting systems are noetherian. *Memo MTP-3*, 1979.

[20] Tobias Nipkow. Higher-order critical pairs. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS '91), Amsterdam, The Netherlands, July 15-18, 1991*, pages 342–349. IEEE Computer Society, 1991.

[21] Stephan Schulz. E–a brainiac theorem prover. *Ai Communications*, 15(2-3):111–126, 2002.

[22] Alexander Steen and Christoph Benzmüller. The higher-order prover leo-iii. In *International Joint Conference on Automated Reasoning*, pages 108–116. Springer, 2018.

[23] Joachim Steinbach. *Termination of rewriting - extensions, comparison and automatic generation of simplification orderings*. PhD thesis, Kaiserslautern University of Technology, Germany, 1994.

[24] Jaco van de Pol. *Termination of Higher-order Rewrite Systems*. PhD thesis, University of Utrecht, 1996.

[25] Akihisa Yamada, Keiichirou Kusakari, and Toshiki Sakabe. Nagoya termination tool. *CoRR*, abs/1404.6626, 2014.

[26] Akihisa Yamada, Keiichirou Kusakari, and Toshiki Sakabe. A unified ordering for termination proving. *Sci. Comput. Program.*, 111:110–134, 2015.