情報工学コース卒業研究報告書

高階書換え系の変換による停止性証明法に関する研究

平成18年2月9日

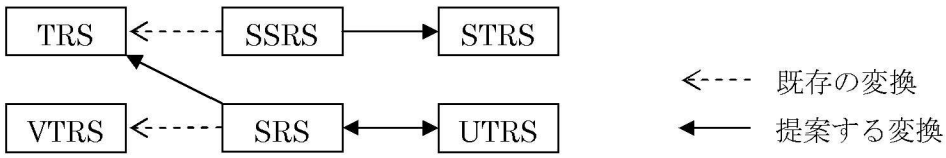
山田 晃久

概要

項書換え系 (Term Rewriting System; TRS) とは、項の書換えの繰り返しにより計算を記述する計算モデルであり、関数型プログラムの停止性検証や定理自動証明などに利用されている。TRS に関する研究は多く、AProVE, Tyrolean Termination Tool, C*i*ME など、検証ツールも充実している。

項書換え系は関数型プログラミングと親和性が高いが、関数型プログラミング言語に一般的に備わっている高階関数を直接表現することができない。そこで、高階関数を扱うために項書換え系を拡張した様々な高階書換え系の枠組みが提案、研究されている。それらは型無し項書換え系 (Untyped TRS; UTRS), 単純型項書換え系 (Simply-typed TRS; STRS), S 式書換え系 (S-expression Rewriting System; SRS), 単純型 S 式書換え系 (Simply-typed SRS; SSRS) などであり、それぞれ研究されている。例えば Aoto, Yamada らは SSRS を TRS に変換して停止性証明を行う方法を示している。また Toyama は、アリティ可変の項書換え系 (Variadic TRS; VTRS) において辞書式経路順序による停止性証明法を示し、さらに SRS を VTRS の特殊形としてとして定式化することでその停止性を証明する方法を提案している。特に STRS では、Kusakari, Sakai らにより強計算依存対に基づく効率のよい停止性証明法 (Strong Computability Dependency Pair Method; SCDP 法) が提案されている。

本研究では様々な書換え系間の変換を提案する。そしてそれらの変換を用いて、各種高階書換え系の停止性を、変換後の書換え系の停止性に帰着させて検証する。これによって既存の手法では困難であった書換え系の停止性を、別の枠組みで提案された手法を用いて検証することができるようになる。下図に本研究で提案した変換を示す。



まず、SRS を TRS に変換する方法を提案し、SRS の停止性証明を変換後の TRS の停止性証明に帰着させる。ただしこの方法では、入力となる SRS に若干の制限が加えられる。しかしこの制限は通常のプログラムを表記する際に問題となるほどのものではない。この方法により、既存の方法では自動証明が困難だったいくつかの SRS の停止性が、TRS 上の辞書式経路順序によって証明することに成功する。

次に、UTRS を SRS に変換する方法を提案し、UTRS の停止性を、SRS、TRS の順に変換することで停止性証明を行うという方法を提案する。しかし変換後に得られる TRS の停止性は証明困難な場合が多いことがわかる。これは型無し項がアリティ一定でないことによるものである。最新の証明法を用いた場合の効果に関しては今後の課題としたい。

最後に、SSRS から STRS への変換を提案し、この変換を用いて SCDP 法を SSRS 上に導入する。証明に必要となる依存対などの概念を、変換後のそれと一致するよう定義することで STRS における結果をそのまま利用することができるようになる。

目 次

第1章　序論

項書換え系 (Term Rewriting System; TRS) とは、項の書換えの繰り返しにより計算を記述する計算モデルであり、関数型プログラムの停止性検証や定理自動証明などに利用されている。TRS に関する研究は多く、AProVE[2], Tyrolean Termination Tool[1], C*i*ME[3] など、検証ツールも充実している。

項書換え系は関数型プログラミングと親和性が高いが、関数型プログラミング言語に一般的に備わっている高階関数を直接表現することができない。

高階関数とは、たとえば Standerd ML で記述された以下に示す map 関数ような関数である。

$$\left\{ \begin{array}{l} \texttt{fun map f [] = []} \\ \texttt{\ \ | \ map f x :: xs = f x :: map f xs;} \end{array} \right.$$

ここで登場する変数 f は右辺では引数をとるため、このような書換え系は TRS では表現できない。

そこで、高階関数を扱うために項書換え系を拡張した様々な高階書換え系の枠組みが提案、研究されている。それらは型無し項書換え系 (Untyped TRS; UTRS), 単純型項書換え系 (Simply-typed TRS; STRS)[7], S 式書換え系 (S-expression Rewriting System; SRS)[11], 単純型 S 式書換え系 (Simply-typed SRS; SSRS) などであり、それぞれ研究されている。例えば上に挙げた map 関数を表す SRS $\mathcal{R}_{map}$ は、

$$\mathcal{R}_{map} = \left\{ \begin{array}{l} (map\ f\ nil) \to nil \\ (map\ f\ (cons\ x\ xs)) \to (cons\ (f\ x)\ (map\ f\ xs)) \end{array} \right.$$

また UTRS $\mathcal{R}'_{map}$ は、

$$\mathcal{R}'_{map} = \left\{ \begin{array}{l} map[f, nil] \to nil \\ map[f, cons[x, xs]] \to cons[f[x], map[f, xs]] \end{array} \right.$$

などと表される。map の型を $(N \to N) \times L \to L$、$cons$ の型を $N \times L \to L$ としたとき、SSRS $\mathcal{R}_{tmap}$ は $\mathcal{R}_{map}$ と同様に、また STRS $\mathcal{R}'_{tmap}$ は、

$$\mathcal{R}'_{tmap} = \begin{cases} map[(f, nil)] \to nil \\ map[(f, cons[(x, xs)])] \to cons[(f[x], map[(f, xs)])] \end{cases}$$

と表される。

Aoto, Yamada らは SSRS と等価である STTRS を TRS に変換して停止性証明を行う方法を示している [5]。

また Toyama は、アリティ可変の項書換え系 (Variadic TRS; VTRS) において辞書式経路順序による停止性証明法を示し、さらに SRS を VTRS の特殊形として定式化することでその停止性を証明する方法を提案している [11]。この手法では、$\mathcal{R}_{map}$ は以下のような VTRS として表される。

$$\mathcal{R}_{map} = \begin{cases} \circ(map, f, nil) \to nil \\ \circ(map, f, \circ(cons, x, xs)) \to \circ(cons, \circ(f, x), \circ(map, f, xs)) \end{cases}$$

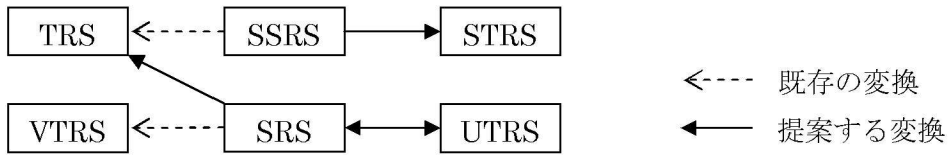
しかしこの VTRS の停止性は辞書式経路順序では示すことができない。

特に STRS では、Kusakari, Sakai らにより強計算依存対に基づく効率のよい停止性証明法 (Strong Computability Dependency Pair Method; SC-DP 法) が提案されている [8]。この手法では SSRS $\mathcal{R}_{tmap}$ の停止性は、以下に示す再帰成分 $RC_{SC}(\mathcal{R}_{tmap})$ を検証することで行われる。

$$RC_{SC}(\mathcal{R}_{tmap}) = \left\{ \{\langle map^\sharp[(f, cons[(x, xs)])], map^\sharp[(f, xs)]\rangle\} \right.$$

SC-DP 法では高階変数を解析する必要がなく、高階変数にために複雑になる関数間の依存関係を解析する労力を大幅に削減することができる。この手法が適用できる PFP という条件は、通常のプログラムを記述する際に問題になるものではない。

本研究ではこれら様々な書換え系間の変換を提案する。そしてそれらの変換を用いて、各種高階書換え系の停止性を、変換後の書換え系の停止性に帰着させて検証する。これによって既存の手法では困難であった書換え系の停止性を、別の枠組みで提案された手法を用いて検証することができるようになる。下図に本研究で提案した変換を示す。



まず第 3 章で、SRS を TRS に変換する方法を提案し、SRS の停止性証明を変換後の TRS の停止性証明に帰着させる。この方法は S 式 $(f\ s_1\ \cdots\ s_n)$ を項 $f(t_1,\cdots,t_n)$ に対応させる。f が関数記号でない場合は [11] の方法と同様 $\circ$ 記号を用いるが、これにアリティ情報を与えることで TRS の枠組みに収めることができる。ただしこの方法では、入力となる SRS に若干の制限が加えられる。しかしこの制限は通常のプログラムを表記する際に問題となるほどのものではない。たとえば、この変換を用いて $\mathcal{R}_{map}$ を変換すると

$$\overline{\mathcal{R}_{map}} \cup \mathcal{A}(\mathcal{R}_{map}) = \begin{cases} map(f, nil') \to nil' \\ map(f, cons(x, xs)) \to cons(\circ^1(f, x), map(f, xs)) \\ \circ^2(map', v_1, v_2) \to map(v_1, v_2) \\ \circ^2(cons', v_1, v_2) \to cons(v_1, v_2) \end{cases}$$

となる。この TRS の停止性は辞書式経路順序によって示すことができる。この方法により、既存の方法では自動証明が困難だったいくつかの SRS の停止性が、TRS 上の辞書式経路順序によって証明することに成功する。

次に第 4 章で、UTRS を SRS に変換する方法を提案し、UTRS の停止性を、SRS、TRS の順に変換することで停止性証明を行うという方法を提案する。たとえば、$\mathcal{R}'_{map}$ を変換して得られる TRS は以下のようになる。

$$\overline{\mathcal{R}'_{map}} = \begin{cases} \circ^1(map(f), nil') \to nil' \\ \circ^1(map(f), \circ^1(cons(x), xs)) \to \circ^1(cons(\circ^1(f, x)), \circ^1(map(f), xs)) \\ \circ^1(map', v_1) \to map(v_1) \\ \circ^1(cons', v_1) \to cons(v_1) \end{cases}$$

しかし変換後に得られる TRS の停止性は証明困難な場合が多いことがわかる。これは型無し項がアリティ一定でないことによるものである。最新の証明法を用いた場合の効果に関しては今後の課題としたい。

最後に第 5 章で、SSRS から STRS への変換を提案し、この変換を用いて SCDP 法を SSRS 上に導入する。たとえば $\mathcal{R}_{tmap}$ を提案する手法で変換すると、STRS

$\mathcal{R}'_{tmap}$ が得られる。SCDP 法を用いた証明に必要となる依存対などの概念を、変換後のそれと一致するよう定義することで STRS における結果をそのまま利用することができるようになる。変換は理論的根拠を与えるだけであり、停止性検証に使用される部分項基準の概念を SSRS 上で定義しなおすことで、実際に変換を行わずに停止性証明を行うことができる。

第2章　準備

2.1　抽象書換え系

複数の書換え系の枠組みを扱うため、書換え系の一般的枠組みである抽象書換え系を、論文 [10] を参考に定義する。

定義 2.1.1 (抽象書換え系) 集合 A、A 上の 2 項関係 $\rightarrow$ について、対 $\mathcal{R} = \langle A, \rightarrow \rangle$ を抽象書換え系 (Abstruct Reduction System; ARS) と呼ぶ。ここで、$\rightarrow$ の反射・推移閉包を $\rightarrow^*$、推移閉包を $\rightarrow^+$ と表す。

定義 2.1.2 (正規形) ARS $\mathcal{R} = \langle A, \rightarrow \rangle, a \in A$ について $a \rightarrow b$ となる b が存在しないとき、a は正規形 (normal form) であるという。また、$a \rightarrow^* b$ となる正規形 b が存在するとき、b は a の正規形であるという。

定義 2.1.3 (停止性) ARS $\mathcal{R} = \langle A, \rightarrow \rangle$ について、$s_1 \rightarrow s_2 \rightarrow \cdots$ となる無限列 $s_1, s_2, \cdots$ が存在しないとき、$\mathcal{R}$ は停止性を持つ (terminating) または強正規性を持つ (strongly normalizing) といい、$SN(\mathcal{R})$ と表す。

2.2　項書換え系

文献 [6] の定義を元に、項書換え系に関する定義を述べる。

定義 2.2.1 (項) $\Sigma \cap \mathcal{V} = \emptyset$ となるような関数集合 Σ、変数集合 $\mathcal{V}$、写像 $arity : \Sigma \rightarrow \mathbb{N}$ によって定義される 1 階の項集合 $\mathcal{T}_1(\Sigma, \mathcal{V})$ を、以下のように帰納的に定義する。

1. $\mathcal{V} \subseteq \mathcal{T}_1(\Sigma, \mathcal{V})$
2. $f(t_1, \cdots, t_n) \in \mathcal{T}_1(\Sigma, \mathcal{V})$ if $f \in \Sigma, n = arity(f)$ and $t_1, \cdots, t_n \in \mathcal{T}_1(\Sigma, \mathcal{V})$

$f()$ を単に f で略記する。項 t に現れる変数の集合を $Var(t)$、関数の集合を $Fun(t)$ で表す。

定義 2.2.2 (代入) 変数集合 $\mathcal{V}$ から項集合 $\mathcal{T}_1(\Sigma, \mathcal{V})$ への写像を代入という。代入 θ を $\mathcal{T}_1(\Sigma, \mathcal{V})$ 上に拡張した写像 $\hat{\theta}$ を以下のように定義する。

1. $\hat{\theta}(v) = \theta(v)$ if $v \in \mathcal{V}$
2. $\hat{\theta}(f(t_1, \cdots, t_n)) = f(\hat{\theta}(t_1), \cdots, \hat{\theta}(t_n))$

定義域が $\{v_1, \cdots, v_n\}$ である代入 θ を、$[v_1 := \theta(v_1), \cdots, v_n := \theta(v_n)]$ とも表す。以降、$\hat{\theta}$ を単に θ で表す。また、$\theta(t)$ を $t\theta$ と表す。

定義 2.2.3 (文脈) ホールという変数 $\Box$ をひとつだけ含む $\mathcal{T}_1(\Sigma, \mathcal{V} \cup \{\Box\})$ 上の項 $C[\,]$ を文脈という。$t \in \mathcal{T}_1(\Sigma, \mathcal{V})$ について、$C[\,]$ 中のホールを t で置き換えた項 $C[\,][\Box := t]$ を $C[t]$ で表す。

定義 2.2.4 (書換え規則) $\mathcal{T}_1(\Sigma, \mathcal{V})$ 上の対 $\langle l, r \rangle$ のうち、以下の条件を満たすものを書換え規則といい、$l \to r$ で表す。

- $l \notin \mathcal{V}$
- $Var(r) \subseteq Var(l)$

定義 2.2.5 (項書換え系) $\mathcal{T}_1(\Sigma, \mathcal{V})$ 上の書換え規則の集合 $\mathcal{R}$ から、書換え関係 $\to_{\mathcal{R}}$ を以下のように定義する。

$$t \to_{\mathcal{R}} u \overset{def}{\iff} \exists C[\,].\, \exists \theta.\, \exists l \to r \in \mathcal{R}.\, t = C[l\theta] \wedge u = C[r\theta]$$

このとき ARS $\langle \mathcal{T}_1(\Sigma, \mathcal{V}), \to_{\mathcal{R}} \rangle$ を、項書換え系 (Term Rewriting System; TRS) という。また、$\mathcal{T}_1(\Sigma, \mathcal{V})$ 上の TRS であることが明らかな場合、$\langle \mathcal{T}_1(\Sigma, \mathcal{V}), \to_{\mathcal{R}} \rangle$ を単に $\mathcal{R}$ で表してよいことにする。

2.3　型無し項書換え系

文献 [8] で定義されている型無し項書換え系に関する定義を述べる。

定義 2.3.1 (型無し項) $\Sigma \cap \mathcal{V} = \emptyset$ となるような関数集合 Σ、変数集合 $\mathcal{V}$ によって定義される型無し項の集合 $\mathcal{T}(\Sigma, \mathcal{V})$ を、以下のように帰納的に定義する。

$$t = a[t_1, \cdots, t_n] \in \mathcal{T}(\Sigma, \mathcal{V}) \text{ if } a \in \Sigma \cup \mathcal{V} \text{ and } t_1, \cdots, t_n \in \mathcal{T}(\Sigma, \mathcal{V})$$

$a[\,]$ を単に a と略記する。a を t の根記号といい、$root(t)$ で、リスト $[t_1, \cdots, t_n]$ を引数といい、$args(t)$ で表す。型無し項 t に現れる変数の集合を $Var(t)$、関数の集合を $Fun(t)$ で表す。また $t = a[t_1, \cdots, t_m]$ について、$a[t_1, \cdots, t_m, u_1, \cdots, u_n]$ を $t[u_1, \cdots, u_n]$ とも表す。

定義 2.3.2 (代入) 変数集合 $\mathcal{V}$ から型無し項集合 $\mathcal{T}(\Sigma, \mathcal{V})$ への写像を代入という。代入 θ を $\mathcal{T}(\Sigma, \mathcal{V})$ 上に拡張した写像 $\hat{\theta}$ を以下のように定義する。

- $\hat{\theta}(f[t_1, \cdots, t_n]) = f[\hat{\theta}(t_1), \cdots, \hat{\theta}(t_n)]$ if $f \in \Sigma$
- $\hat{\theta}(v[t_1, \cdots, t_n]) = a[u_1, \cdots, u_m, \hat{\theta}(t_1), \cdots, \hat{\theta}(t_n)]$
  if $v \in \mathcal{V}$ and $\theta(v) = a[u_1, \cdots, u_m]$

定義域が $\{v_1, \cdots, v_n\}$ である代入 θ を、$[v_1 := \theta(v_1), \cdots, v_n := \theta(v_n)]$ とも表す。以降、$\hat{\theta}$ を単に θ で表す。また、$\theta(t)$ を $t\theta$ と表す。

定義 2.3.3 (位置) 型無し項 t 中の位置の集合 $Pos(t) \subseteq \mathbb{N}^*$ を、以下のように帰納的に定義する。

1. $Pos(a) = \{\varepsilon\}$ if $a \in \Sigma \cup \mathcal{V}$
2. $Pos(a[t_1, \cdots, t_n]) = \{\varepsilon\} \cup \bigcup_{i=1}^{n} \{ip \mid p \in Pos(t_i)\}$

位置 ε を根位置という。葉位置とは、$p \in Pos(s) \wedge p1 \notin Pos(s)$ を満たす位置 p のことをいう。また、位置 p における部分項 $t|_p$ を、以下のように帰納的に定義する。

1. $t|_\varepsilon = t$
2. $a[t_1, \cdots, t_n]|_{ip} = t_i|_p$ for $i \in \mathbb{N}$

定義 2.3.4 (文脈) ホールという変数 $\Box$ をひとつだけ含む $\mathcal{T}(\Sigma, \mathcal{V} \cup \{\Box\})$ 上の項 $C[\,]$ を文脈という。特に、$\Box$ が葉位置に現れる文脈を葉文脈、根記号が $\Box$ であるような文脈を根文脈という。$t \in \mathcal{T}(\Sigma, \mathcal{V})$ について、$C[\,]$ 中のホールを t で置き換えた項 $C[\,][\Box := t]$ を $C[t]$ で表す。

定義 2.3.5 (書換え規則) $\mathcal{T}(\Sigma, \mathcal{V})$ 上の対 $\langle l, r\rangle$ のうち、以下の条件を満たすものを書換え規則といい、$l \to r$ で表す。

- $root(l) \notin \mathcal{V}$
- $Var(r) \subseteq Var(l)$

定義 2.3.6 (型無し項書換え系) $\mathcal{T}(\Sigma, \mathcal{V})$ 上の書換え規則の集合 $\mathcal{R}$ から、書換え関係 $\to_{\mathcal{R}}$ を以下のように定義する。

$$t \to_{\mathcal{R}} u \iff^{def} \exists C[\,].\, \exists \theta.\, \exists l \to r \in \mathcal{R}.\, t = C[l\theta] \wedge u = C[r\theta]$$

このとき ARS $\langle \mathcal{T}(\Sigma, \mathcal{V}), \to_{\mathcal{R}}\rangle$ を、型無し項書換え系 (Untyped Term Rewriting System; UTRS) という。また、$\mathcal{T}(\Sigma, \mathcal{V})$ 上の UTRS であることが明らかな場合、$\langle \mathcal{T}(\Sigma, \mathcal{V}), \to_{\mathcal{R}}\rangle$ を単に $\mathcal{R}$ で表してよいことにする。

定義 2.3.7 (被定義記号と構成子記号) UTRS $\mathcal{R}$ について、被定義記号の集合 $\mathcal{D}_{\mathcal{R}}$ および構成子記号の集合 $\mathcal{C}_{\mathcal{R}}$ を、以下のように定義する。

$$\mathcal{D}_{\mathcal{R}} = \{root(l) \mid l \to r \in \mathcal{R}\}$$

$$\mathcal{C}_{\mathcal{R}} = \Sigma - \mathcal{D}_{\mathcal{R}}$$

2.4 S式書換え系

文献 [11] では S 式は項の特別な形として定義される。しかし本論文では S 式および S 式書換え系の定義を独立に与え、これを変換する形で TRS と結びつけることにする。

定義 2.4.1 (S式) $\Sigma \cap \mathcal{V} = \emptyset$ となるような定数集合 Σ、変数集合 $\mathcal{V}$ によって定義される S 式の集合 $\mathcal{S}(\Sigma, \mathcal{V})$ を、以下の様に帰納的に定義する。

1. $\Sigma \cup \mathcal{V} \subseteq \mathcal{S}(\Sigma, \mathcal{V})$
2. $(s_0 \ \cdots \ s_n) \in \mathcal{S}(\Sigma, \mathcal{V})$

S 式 s に現れる変数の集合を $Var(s)$、関数の集合を $Fun(s)$ で表す。

定義 2.4.2 (根記号) S 式における根記号を、以下のように帰納的に定義する。

1. $root(a) = a$ if $a \in \Sigma \cup \mathcal{V}$
2. $root((s_0\ \cdots\ s_n)) = root(s_0)$

定義 2.4.3 (代入) 変数集合 $\mathcal{V}$ から S 式 $\mathcal{S}(\Sigma, \mathcal{V})$ への写像を代入という。代入 θ を $\mathcal{S}(\Sigma, \mathcal{V})$ 上に拡張した $\hat{\theta}$ を以下のように定義する。

1. $\hat{\theta}(v) = \theta(v)$ if $v \in \mathcal{V}$
2. $\hat{\theta}(c) = c$ if $c \in \Sigma$
3. $\hat{\theta}(\,(s_1\ \cdots\ s_n)\,) = (\,\hat{\theta}(s_1)\ \cdots\ \hat{\theta}(s_n)\,)$

定義域が $\{v_1, \cdots, v_n\}$ である代入 θ を、$[v_1 := \theta(v_1), \cdots, v_n := \theta(v_n)]$ とも表す。以降、$\hat{\theta}$ を単に θ で表す。また、$\theta(s)$ を $s\theta$ で表す。

定義 2.4.4 (S 式における位置) S 式 s 中の位置の集合 $S\_Pos(s) \subseteq \mathbb{N}^*$ を、以下のように帰納的に定義する。

1. $S\_Pos(a) = \{\varepsilon\}$ if $a \in \Sigma \cup \mathcal{V}$
2. $S\_Pos((s_0\ \cdots\ s_n)) = \{\varepsilon\} \cup \bigcup_{i=0}^{n} \{ip \mid p \in S\_Pos(s_i)\}$

$S\_Pos(s)$ 中の最長の 0 列を根位置という。$pi \in S\_Pos(s)$ が葉位置であるとは、$i \neq 0 \wedge pi0 \notin S\_Pos(s)$ を満たすことをいう。

定義 2.4.5 (文脈) ホールと呼ぶ特別な変数 $\Box \notin \mathcal{V}$ をひとつだけ含む $\mathcal{S}(\Sigma, \mathcal{V} \cup \{\Box\})$ 上の S 式を文脈といい、$C[\,]$ などと表す。特に、$\Box$ が葉位置に現れる文脈を葉文脈、根記号が $\Box$ であるような文脈を根文脈という。また、$C[\,]$ 中のホールに s を代入した S 式 $C[\,][\Box := s]$ を $C[s]$ と表す。

定義 2.4.6 (書換え規則) $l, r \in \mathcal{S}(\Sigma, \mathcal{V})$ について、以下の性質を満たす対 $\langle l, r \rangle$ を書き換え規則と呼び、$l \to r$ と表す。

1. $root(l) \notin \mathcal{V}$
2. $Var(r) \subseteq Var(l)$

またこの規則から定義される書換え関係を以下のように定義する。

定義 2.4.7 (S 式書換え系) 書き換え規則の集合 $\mathcal{R}$ から簡約関係 $\to_{\mathcal{R}}$ を以下のように定義する。

$$s \to_{\mathcal{R}} t \overset{def}{\iff} \exists C[\,].\, \exists \theta.\, \exists l \to r \in \mathcal{R}.\, s = C[l\theta] \wedge t = C[r\theta]$$

このとき ARS $\langle \mathcal{S}(\Sigma, \mathcal{V}), \rightarrow_{\mathcal{R}} \rangle$ を、S 式書き換え系 (S-Expression Rewriting System; SRS) と呼ぶ。また、SRS であることが明らかな場合、$\langle \mathcal{S}(\Sigma, \mathcal{V}), \rightarrow_{\mathcal{R}} \rangle$ を単に $\mathcal{R}$ で表してよいことにする。

定義 2.4.8 (被定義記号と構成子記号) SRS $\mathcal{R}$ について、被定義記号の集合 $\mathcal{D}_{\mathcal{R}}$ および構成子記号の集合 $\mathcal{C}_{\mathcal{R}}$ を、以下のように定義する。

$$\mathcal{D}_{\mathcal{R}} = \{root(l) \mid l \rightarrow r \in \mathcal{R}\}$$

$$\mathcal{C}_{\mathcal{R}} = \Sigma - \mathcal{D}_{\mathcal{R}}$$

第3章　S式書換え系の変換

まず本論文で議論する書換え系の変換について一般に成り立つ補題を与える。

3.1　停止性証明に使える変換の性質

定義 3.1.1 ARS $\mathcal{R}_1 = \langle A_1, \to_1 \rangle, \mathcal{R}_2 = \langle A_2, \to_2 \rangle$ について、写像 $\phi : A_1 \to A_2$ が存在して、すべての $s, t \in A_1$ について

$$s \to_1^+ t \implies \phi(s) \to_2^+ \phi(t)$$

となるとき、R_2 は R_1 をシミュレートする、という。

定理 3.1.2 ARS $\mathcal{R}_1, \mathcal{R}_2$ について、$\mathcal{R}_2$ が $\mathcal{R}_1$ をシミュレートするならば以下の関係が成り立つ。

1. $SN(\mathcal{R}_1, a) \impliedby SN(\mathcal{R}_2, \phi(a))$
2. $SN(\mathcal{R}_1) \impliedby SN(\mathcal{R}_2)$

(証明) $\mathcal{R}_1 = \langle A_1, \to_1 \rangle, \mathcal{R}_2 = \langle A_2, \to_2 \rangle$ とおく。また $\neg SN(\mathcal{R}_1, a)$ と仮定する。すなわち無限の書換え系列 $a \to_1^+ a_1 \to_1^+ a_2 \to_1^+ \cdots$ が存在する。このとき写像 $\phi : A_1 \to A_2$ が存在して、

$$\phi(a) \to_2^+ \phi(a_1) \to_2^+ \phi(a_2) \to_2^+ \cdots$$

なる無限書換え系列が存在するので、

$$\neg SN(\mathcal{R}_2, a)$$

となる。この対偶をとれば 1 つ目の関係は示される。これから、明らかに 2 つ目の関係が導かれる。　□

定義 3.1.3 ARS $\langle A, \to \rangle$ について、写像の集合 $\Theta \subseteq A^A, \Delta \subseteq A^A$、関係 $\mathcal{R} \subseteq A^2$ が存在して以下の式が成立するとき、$\langle A, \to \rangle$ を $\langle A, \mathcal{R}, \Theta, \Delta \rangle$ で表す。

$$a \to b \iff \exists \langle l, r \rangle \in \mathcal{R}.\, \exists C \in \Delta.\, \exists \theta \in \Theta.\, a = C(\theta(l)) \wedge b = C(\theta(r))$$

補題 3.1.4 ARS $\langle A_1, \to_{\mathcal{R}} \rangle = \langle A_1, \mathcal{R}, \Theta_1, \Delta_1 \rangle$、写像 $\phi : A_1 \to A_2$ を考える。ARS $\langle A_2, \to_{\mathcal{Q}} \rangle = \langle A_2, \mathcal{Q}, \Theta_2, \Delta_2 \rangle$ が存在して、任意の $\langle l, r \rangle \in \mathcal{R}$ について以下の条件が成立するとする。

- $\forall \theta_1 \in \Theta_1.\, \exists \theta_2 \in \Theta_2.\, \phi(\theta_1(l)) = \theta_2(\phi(l)) \wedge \theta_2(\phi(r)) \to_{\mathcal{Q}}^* \phi(\theta_1(r))$ かつ、
- $\forall C_1 \in \Delta_1.\, \exists C_2 \in \Delta_2.\, \phi(C_1(l)) = C_2(\phi(l)) \wedge C_2(\phi(r)) \to_{\mathcal{Q}}^* \phi(C_1(r))$

このとき、ARS $\langle A_2, \phi(\mathcal{R}) \cup \mathcal{Q}, \Theta_2, \Delta_2 \rangle$ は ARS $\langle A_1, \mathcal{R}, \Theta_1, \Delta_1 \rangle$ をシミュレートする。

(証明) $a \to_{\mathcal{R}} b$ とする。すなわち $\theta_1 \in \Theta_1, C_1 \in \Delta_1, \langle l, r \rangle \in \mathcal{R}$ が存在して、

$$a = C_1(\theta_1(l)) \text{ and } b = C_1(\theta_1(r))$$

仮定より $\theta_2 \in \Theta_2, C_2 \in \Delta_2$ が存在して、

$$\phi(a) = C_2(\theta_2(\phi(l)))$$

かつ、

$$\begin{aligned} C_2(\theta_2(\phi(r))) &\to_{\mathcal{Q}}^* C_2(\phi(\theta_1(r))) \\ &\to_{\mathcal{Q}}^* \phi(C_1(\theta_1(r))) = \phi(b) \end{aligned}$$

定義より $\langle \phi(l), \phi(r) \rangle \in \phi(\mathcal{R})$ なので、

$$\phi(a) = C_2(\theta_2(\phi(l))) \to_{\phi(\mathcal{R})} C_2(\theta_2(\phi(r))) \to_{\mathcal{Q}}^* \phi(b)$$

したがって、

$$\phi(a) \to_{\phi(\mathcal{R}) \cup \mathcal{Q}}^+ \phi(b)$$

□

3.2 SRS から TRS への変換

定義 3.2.1 SRS $\mathcal{R}$ における仮名の集合 $aliases(\mathcal{R})$、本名の集合 $names(\mathcal{R})$ を以下のように定義する。

$$\begin{aligned} aliases(\mathcal{R}) &= \{f \in \Sigma \mid \exists r.\, f \to r \in \mathcal{R}\} \\ names(\mathcal{R}) &= \Sigma - aliases(\mathcal{R}) \end{aligned}$$

SRS 上の関数集合 Σ に対する TRS 上の関数集合 Σ' を以下のように定義する。

$$\Sigma' = \Sigma \cup \{f' \mid f \in \Sigma\} \cup \{\circ^n \mid n \in \mathbb{N}\} \cup \{f^n \mid f \in \Sigma, n \in \mathbb{N}\}$$

定義 3.2.2 (S 式から項への変換) $\mathcal{S}(\Sigma, \mathcal{V})$ 上の SRS $\mathcal{R}$ について、$\mathcal{S}(\Sigma, \mathcal{V})$ から $\mathcal{T}_1(\Sigma', \mathcal{V})$ への変換 ϕ^{trs}_{srs} を以下のように定義する。ただし、

1. $\phi^{trs}_{srs}(v) = v$ if $v \in \mathcal{V}$
2. $\phi^{trs}_{srs}(f) = f'$ if $f \in \Sigma$
3. $\phi^{trs}_{srs}((f\ s_1\ \cdots\ s_n)) = f^n(\phi^{trs}_{srs}(s_1), \cdots, \phi^{trs}_{srs}(s_n))$ if $f \in names(\mathcal{R})$
4. $\phi^{trs}_{srs}((s_0\ \cdots\ s_n)) = \circ^n(\phi^{trs}_{srs}(s_0), \cdots, \phi^{trs}_{srs}(s_n))$ otherwise

関数記号の肩の n は引数の個数から明らかなので、混乱のない場合は省略することにする。

定義 3.2.3 代入 $\theta : \mathcal{V} \to \mathcal{S}(\Sigma, \mathcal{V})$、文脈 C、規則集合 $\mathcal{R}$ の変換を以下のように定義する。

1. $\phi^{trs}_{srs}(\theta)(v) = \phi^{trs}_{srs}(\theta(v))$ for $v \in \mathcal{V}$
2. $\phi^{trs}_{srs}(C)[t] = \phi^{trs}_{srs}(C[\,])[\Box := t]$
3. $\phi^{trs}_{srs}(\mathcal{R}) = \{\phi^{trs}_{srs}(l) \to \phi^{trs}_{srs}(r) \mid l \to r \in \mathcal{R}\}$

本節では S 式、SRS、文脈または代入 x について、$\overline{x} = \phi^{trs}_{srs}(x)$ と略記する。

このままでは引数を取る変数への代入 θ に対して、$\overline{s}\sigma = \overline{s\theta}$ となる項上の代入 σ が存在しない。そこで $\overline{s}\,\overline{\theta} \to^*_{\mathcal{A}(\mathcal{R})} \overline{s\theta}$ となるような TRS $\mathcal{A}(\mathcal{R})$ を定義する。

定義 3.2.4

$$\begin{aligned} \mathcal{A}(\mathcal{R}) = \{\circ^n(f', v_1, \cdots, v_n) \to f^n(v_1, \cdots, v_n) \mid f \in names(\mathcal{R}), \\ \exists C[\,].\, \exists s_1, \cdots, s_n, r.\, C[(f\ s_1\ \cdots\ s_n)] \to r \in \mathcal{R}\} \end{aligned}$$

補題 3.2.5 S式 $s \in \mathcal{S}(\Sigma, \mathcal{V})$ が引数を取る変数を持たないか、代入 θ が $\forall v.\theta(v) \notin names(\mathcal{R})$ であるとき、

$$\overline{s}\,\overline{\theta} = \overline{s\theta}$$

(証明) 構造再帰で示す。

基底段階: $s \in \Sigma \cup \mathcal{V}$ について、

1. $s = f \in \Sigma$ のとき、$\overline{s}\,\overline{\theta} = f'\overline{\theta} = f' = \overline{f\theta}$
2. $s = v \in \mathcal{V}$ のとき、$\overline{s}\,\overline{\theta} = v\overline{\theta} = \overline{v\theta}$

したがって、$\overline{s}\,\overline{\theta} = \overline{s\theta}$

帰納段階: $\overline{s_i}\,\overline{\theta} = \overline{s_i\theta}$ と仮定する。$s = (s_0\ s_1\ \cdots\ s_n)$ について、

$$\begin{aligned}
\overline{s}\,\overline{\theta} &= \circ(\overline{s_0}, \cdots, \overline{s_n})\overline{\theta} && \text{(変換の定義より)}\\
&= \circ(\overline{s_0}\,\overline{\theta}, \cdots, \overline{s_n}\,\overline{\theta}) && \text{(代入の定義より)}\\
&= \circ(\overline{s_0\theta}, \cdots, \overline{s_n\theta}) && \text{(帰納法の仮定より)}\\
&= \overline{(s_0\theta\ \cdots\ s_n\theta)} && (s_0\theta \notin names(\mathcal{R}) \text{ の仮定より})\\
&= \overline{s\theta}
\end{aligned}$$

したがって、条件を満たすすべての s, θ について、$\overline{s}\,\overline{\theta} = \overline{s\theta}$ □

補題 3.2.6 $s \in \mathcal{S}(\Sigma, \mathcal{V}) - names(\mathcal{R})$ について、

$$\overline{C}[\overline{s}] = \overline{C[s]}$$

(証明)

$$\begin{aligned}
\overline{C}[\overline{s}] &= \overline{C[\,]}[\Box := \overline{s}] && \text{(文脈の変換の定義より)}\\
&= \overline{C[\,]}\,\overline{[\Box := s]} && \text{(代入の変換の定義より)}\\
&= \overline{C[\,][\Box := s]} && \text{(補題 3.2.5 より)}\\
&= \overline{C[s]} && \text{(文脈の定義より)}
\end{aligned}$$

□

補題 3.2.7 $s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、

$$\overline{s}\,\overline{\theta} \to^*_{\mathcal{A}(\mathcal{R})} \overline{s\theta}$$

(証明) 構造再帰で示す。

基底段階: $s \in \Sigma \cup \mathcal{V}$ の場合は補題 3.2.5 と同様。

帰納段階: $\overline{s_i}\,\overline{\theta} \to^*_{\mathcal{A}(\mathcal{R})} \overline{s_i\theta}$ と仮定する。$s = (s_0\ s_1\ \cdots\ s_n)$ について、

1. $s_0\theta \notin names(\mathcal{R})$ のときは補題 3.2.5 と同様。
2. $s_0 = v$ かつ $v\theta = f \in names(\mathcal{R})$ のとき、

$$\begin{array}{rcll}
\overline{s}\,\overline{\theta} & = & \circ(\overline{v}, \overline{s_1}, \cdots, \overline{s_n})\,\overline{\theta} & \text{(変換の定義より)} \\
& = & \circ(\overline{v}\,\overline{\theta}, \overline{s_1}\,\overline{\theta}, \cdots, \overline{s_n}\,\overline{\theta}) & \text{(代入の定義より)} \\
& \to^*_{\mathcal{A}(\mathcal{R})} & \circ(\overline{v\theta}, \overline{s_1\theta}, \cdots, \overline{s_n\theta}) & \text{(帰納法の仮定より)} \\
& = & \circ(\overline{f}, \overline{s_1\theta}, \cdots, \overline{s_n\theta}) & \text{(代入の変換の定義より)} \\
& = & \circ(f', \overline{s_1\theta}, \cdots, \overline{s_n\theta}) & \text{(変換の定義より)} \\
& \to_{\mathcal{A}(\mathcal{R})} & f(\overline{s_1\theta}, \cdots, \overline{s_n\theta}) & (\mathcal{A}(\mathcal{R}) \text{ の定義より}) \\
& = & \overline{(f\ s_1\theta\ \cdots\ s_n\theta)} & \text{(代入の定義より)} \\
& = & \overline{s\theta} &
\end{array}$$

したがって、任意の $s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、$\overline{s}\,\overline{\theta} \to^*_{\mathcal{A}(\mathcal{R})} \overline{s\theta}$ □

補題 3.2.8 任意の $s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、

$$\overline{C}[\overline{s}] \to^*_{\mathcal{A}(\mathcal{R})} \overline{C[s]}$$

(証明) 補題 3.2.7 より、補題 3.2.6 と同様にして示せる。 □

補題 3.2.9 左辺に、引数をとる変数が現れないような SRS $\mathcal{R}$ について、TRS $\overline{\mathcal{R}} \cup \mathcal{A}(\mathcal{R})$ は $\mathcal{R}$ をシミュレートする。

(証明) 仮定および補題 3.2.5、$l \notin names(\mathcal{R})$ および補題 3.2.6 より、

$$\forall l \to r \in \mathcal{R}.\ \overline{C[l\theta]} = \overline{C}[\overline{l}\,\overline{\theta}]$$

また、補題 3.2.7、補題 3.2.8 より、

$$\forall l \to r \in \mathcal{R}.\ \overline{C}[\overline{r}\,\overline{\theta}] \to^*_{\mathcal{A}(\mathcal{R})} \overline{C[r\theta]}$$

したがって、補題 3.1.4 より示される。 □

定理 3.2.10 左辺に、引数をとる変数が現れないような $\mathcal{R}$ について、TRS $\overline{\mathcal{R}} \cup \mathcal{A}(\mathcal{R})$ が停止するなら $\mathcal{R}$ は停止する。

3.3　SRS から UTRS への変換

定義 3.3.1 $\mathcal{S}(\Sigma,\mathcal{V})$ から $\mathcal{T}(\Sigma\cup\{\bot\},\mathcal{V})$ への変換 ϕ^{utrs}_{srs} を以下のように定義する。

1. $\phi^{utrs}_{srs}(a)=a$ if $a\in\mathcal{V}\cup\Sigma$
2. $\phi^{utrs}_{srs}((s))=\phi^{utrs}_{srs}(s)[\bot]$
3. $\phi^{utrs}_{srs}((s_0\ s_1\ \cdots\ s_n))=\phi^{utrs}_{srs}(s_0)[\phi^{utrs}_{srs}(s_1),\cdots,\phi^{utrs}_{srs}(s_n)]$

定義 3.3.2 代入 $\theta:\mathcal{V}\to\mathcal{S}(\Sigma,\mathcal{V})$、文脈 C、規則集合 $\mathcal{R}$ の変換を以下のように定義する。

1. $\phi^{utrs}_{srs}(\theta)(v)=\phi^{utrs}_{srs}(\theta(v))$ for $v\in\mathcal{V}$
2. $\phi^{utrs}_{srs}(C)[t]=\phi^{utrs}_{srs}(C[\,])[\Box:=t]$
3. $\phi^{utrs}_{srs}(\mathcal{R})=\{\phi^{utrs}_{srs}(l)\to\phi^{utrs}_{srs}(r)\mid l\to r\in\mathcal{R}\}$

本節では S 式、SRS、文脈または代入 x について、$\overline{x}=\phi^{utrs}_{srs}(x)$ と略記する。

補題 3.3.3 任意の S 式 $s\in\mathcal{S}(\Sigma,\mathcal{V})$ および代入 θ について、

$$\overline{s\theta}=\overline{s}\,\overline{\theta}$$

(証明) 構造帰納法で示す。

基底段階: $a\in\Sigma\cup\mathcal{V}$ について、

$$\overline{a\theta}=a\overline{\theta}=\overline{a}\,\overline{\theta}$$

帰納段階: $\overline{s_i\theta}=\overline{s_i}\,\overline{\theta}$ for $i=0,\cdots,n$ と仮定する。このとき、$s=(s_0\ s_1\ \cdots\ s_n)$ について、

$$\begin{aligned}
\overline{(s_0\ s_1\ \cdots\ s_n)\theta} &= \overline{(s_0\theta\ s_1\theta\ \cdots\ s_n\theta)} && \text{(代入の定義より)}\\
&= \overline{s_0\theta}[\overline{s_1\theta},\cdots,\overline{s_n\theta}] && \text{(変換の定義より)}\\
&= \overline{s_0}\,\overline{\theta}[\overline{s_1}\,\overline{\theta},\cdots,\overline{s_n}\,\overline{\theta}] && \text{(仮定より)}\\
&= \overline{s_0}[\overline{s_1},\cdots,\overline{s_n}]\overline{\theta} && \text{(代入の定義より)}\\
&= \overline{(s_0\ s_1\ \cdots\ s_n)}\,\overline{\theta} && \text{(変換の定義より)}
\end{aligned}$$

したがって、任意の S 式 s について、$\overline{s\theta}=\overline{s}\,\overline{\theta}$　□

補題 3.3.4 $s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、

$$\overline{C}[\overline{s}] = \overline{C[s]}$$

(証明)

$$\begin{array}{rcll} \overline{C}[\overline{s}] & = & \overline{C[\,]}[\Box := \overline{s}] & \text{(文脈の変換の定義より)} \\ & = & \overline{C[\,]}\,\overline{[\Box := s]} & \text{(代入の変換の定義より)} \\ & = & \overline{C[\,][\Box := s]} & \text{(補題 3.3.3 より)} \\ & = & \overline{C[s]} & \text{(文脈の定義より)} \end{array}$$

□

定理 3.3.5 任意の SRS $\mathcal{R}$ について、UTRS $\overline{\mathcal{R}}$ は $\mathcal{R}$ をシミュレートする。

(証明) 補題 3.3.4、補題 3.3.3 および補題 3.1.4 によって示される。 □

例 3.3.6 $\mathcal{R}_{map}$ を変換した UTRS $\mathcal{R}'_{map}$ は、以下のようになる。

$$\mathcal{R}'_{map} = \overline{\mathcal{R}_{map}} = \left\{ \begin{array}{l} map[f, nil] \to nil \\ map[f, cons[x, xs]] \to cons[f[x], map[f, xs]] \end{array} \right.$$

3.4 証明例

提案した SRS から TRS への変換を用いた、辞書式経路順序 [9] による停止性証明の例を示す。$\mathcal{R}' = \phi^{trs}_{srs}(\mathcal{R}) \cup \mathcal{A}(\mathcal{R})$ とする。

例 3.4.1 (map)

$$\mathcal{R}_{map} = \left\{ \begin{array}{l} (map\ f\ nil) \to nil \\ (map\ f\ (cons\ x\ xs)) \to (cons\ (f\ x)\ (map\ f\ xs)) \end{array} \right.$$

$$\mathcal{R}'_{map} = \left\{ \begin{array}{l} map(f, nil') \to nil' \\ map(f, cons(x, xs)) \to cons(\circ_1(f, x), map(f, xs)) \\ \circ^2(map', v_1, v_2) \to map(v_1, v_2) \\ \circ^2(cons', v_1, v_2) \to cons(v_1, v_2) \end{array} \right.$$

変換後の TRS の停止性は、$\circ^2 > map > cons, map \geq \circ^1$ とした辞書式経路順序によって示すことができる。

例 3.4.2 (add)

$$\mathcal{R}_{add} = \begin{cases} (Add\ x\ 0) \to 0 \\ (Add\ x\ (S\ y)) \to (S\ (Add\ x\ y)) \end{cases}$$

$$\mathcal{R}'_{add} = \begin{cases} Add(x, 0') \to 0' \\ Add(x, S(y)) \to S(Add(x, y)) \\ \circ^2(Add', x, y) \to Add(x, y) \\ \circ^1(S', x) \to S(x) \end{cases}$$

この停止性は $\circ^2 > Add > S, \circ^1 > S$ とした辞書式経路順序によって示すことができる。

例 3.4.3 (foldl) $\mathcal{R}_{foldl}$ は左畳み込みを行う SRS である。辞書式経路順序がつけられるよう引数並べている。

$$\mathcal{R}_{foldl} = \begin{cases} (Foldl\ f\ nil\ y) \to y \\ (Foldl\ f\ (cons\ x\ xs)\ y) \to (Foldl\ f\ xs\ (f\ y\ x)) \end{cases}$$

$$\mathcal{R}'_{foldl} = \begin{cases} Foldl(f, nil', y) \to y \\ Foldl(f, cons(x, xs), y) \to Foldl(f, xs, \circ^2(f, y, x)) \\ \circ^3(Foldl', f, xs, y) \to Foldl(f, xs, y) \\ \circ^2(cons', x, xs) \to cons(x, xs) \end{cases}$$

この停止性は $\circ^3 > Foldl > \circ^2 > cons$ とした辞書式経路順序で示すことができる。

例 3.4.4 (foldr) $\mathcal{R}_{foldr}$ は右畳み込みを行う SRS である。

$$\mathcal{R}_{foldr} = \begin{cases} (Foldr\ f\ i\ nil) \to x \\ (Foldr\ f\ i\ (cons\ x\ xs)) \to (f\ x\ (Foldr\ f\ i\ xs)) \end{cases}$$

$$\mathcal{R}'_{foldr} = \begin{cases} Foldr(f, i, nil') \to i \\ Foldr(f, i, cons(x, xs)) \to \circ^2(f, x, Foldr(f, i, xs)) \\ \circ^3(Foldr', f, i, xs) \to Foldr(f, i, xs) \\ \circ^2(cons', x, xs) \to cons(x, xs) \end{cases}$$

この停止性は $\circ^3 > Foldr > \circ^2 > cons$ とした辞書式経路順序で示すことができる。

例 3.4.5 (sum)

$$\mathcal{R}_{sum} = \mathcal{R}_{add} \cup \mathcal{R}_{foldr} \cup \{(Sum\ xs) \to (Foldr\ Add\ 0\ xs)\}$$

$$\mathcal{R}'_{sum} = \mathcal{R}'_{add} \cup \mathcal{R}'_{foldr} \cup \begin{cases} Sum(xs) \to Foldr(Add, 0, xs) \\ \circ^1(Sum', xs) \to Sum(xs) \end{cases}$$

この停止性は $\circ^1 > Sum > Foldr, \circ^3 > Foldr > \circ^2 > cons, \circ^2 > Add > S$ とした辞書式経路順序で示すことができる。

例 3.4.6 (if)

$$\mathcal{R}_{if} = \begin{cases} (If\ True\ x\ y) \to x \\ (If\ False\ x\ y) \to y \end{cases}$$

$$\mathcal{R}'_{if} = \begin{cases} If(True', x, y) \to x \\ If(False', x, y) \to y \\ \circ^3(If', b, x, y) \to If(b, x, y) \end{cases}$$

この停止性は $\circ^3 > If$ とした辞書式経路順序で示すことができる。

例 3.4.7 (geq)

$$\mathcal{R}_{geq} = \begin{cases} (Geq\ x\ 0) \to True \\ (Geq\ 0\ (S\ y)) \to False \\ (Geq\ (S\ x)(S\ y)) \to (Geq\ x\ y) \end{cases}$$

$$\mathcal{R}'_{geq} = \begin{cases} Geq(x, 0') \to True' \\ Geq(0', S(y)) \to False' \\ Geq(S(x), S(y)) \to Geq(x, y) \\ \circ^2(Geq', x, y) \to Geq(x, y) \\ \circ^1(S', x) \to S(x) \end{cases}$$

この停止性は $\circ^2 > Geq > True', Geq > False', \circ^1 > S$ とした辞書式経路順序で示すことができる。

例 3.4.8 (insert)

$$\mathcal{R}_{insert} = \mathcal{R}_{if} \cup \mathcal{R}_{geq} \cup \left\{ \begin{array}{l} (Insert\ x\ nil) \to (cons\ x\ nil) \\ (Insert\ x\ (cons\ y\ ys)) \to \\ \ (If \quad (Geq\ x\ y) \\ \qquad\ (cons\ x\ (cons\ y\ ys)) \\ \qquad\ (cons\ y\ (Insert\ x\ ys))) \end{array} \right.$$

$$\mathcal{R}'_{insert} = \mathcal{R}'_{if} \cup \mathcal{R}'_{geq} \cup \left\{ \begin{array}{l} Insert(x, nil') \to cons(x, nil') \\ Insert(x, cons(y, ys)) \to \\ \ If(\quad Geq(x, y), \\ \qquad\ cons(x, cons(y, ys)), \\ \qquad\ cons(y, Insert(x, ys))) \\ \circ^2(Insert', x, ys) \to Insert(x, ys) \\ \circ^2(cons', x, xs) \to cons(x, xs) \end{array} \right.$$

この停止性は $\circ^2 > Insert > If, Insert > cons, \circ^3 > If, Insert \geq Geq > true', false'$ とした辞書式経路順序で示すことができる。

例 3.4.9 (sort)

$$\mathcal{R}_{sort} = \mathcal{R}_{insert} \cup \mathcal{R}_{foldr} \cup \{(Sort\ xs) \to (Foldr\ Insert\ nil\ xs)\}$$

$$\mathcal{R}'_{sort} = \mathcal{R}'_{insert} \cup \mathcal{R}'_{foldr} \cup \left\{ \begin{array}{l} Sort(xs) \to Foldr(Insert', nil', xs) \\ \circ^1(Sort', xs) \to Sort(xs) \end{array} \right.$$

この停止性は、$\circ^1 > Sort > Foldr > \circ^2 > Insert > If, \circ^3 > Foldr, Insert \geq Geq > true', Geq > false', Insert > cons$ とした辞書式経路順序で示すことができる。

本手法の特徴は、項数 n の高階変数を $\circ^n$ で代表させて順序付ける点にあるといえるだろう。このことは逆に、高階変数を持つ規則について、その変数と同じ項数をもつ関数との関係を解析する困難を生じる。例を示す。

例 3.4.10 $\mathcal{R}_{map} \cup \mathcal{R}_{sort}$ の停止性は辞書式経路順序では示せない。$\mathcal{R}_{map}$ を順序付けるためには $\circ^2 > \circ^1$、$\mathcal{R}_{sort}$ は $\circ^1 > \circ^2$ を要求するためである。ただしこの例は AProVE を用いて検証したところ、停止性が示された。

第4章　型無し項書換え系の変換

4.1　UTRSからSRSへの変換

定義 4.1.1 $\mathcal{T}(\Sigma,\mathcal{V})$ から $\mathcal{S}(\Sigma,\mathcal{V})$ への変換 ϕ^{srs}_{utrs} を以下のように定義する。

1. $\phi^{srs}_{utrs}(a) = a$ if $a \in \mathcal{V} \cup \Sigma$
2. $\phi^{srs}_{utrs}(t[u]) = (\ \phi^{srs}_{utrs}(t)\ \phi^{srs}_{utrs}(u)\ )$ if $t, u \in \mathcal{T}(\Sigma,\mathcal{V})$

定義 4.1.2 代入 $\theta : \mathcal{V} \to \mathcal{T}(\Sigma,\mathcal{V})$、文脈 C、規則集合 $\mathcal{R}$ の変換を以下のように定義する。

1. $\phi^{srs}_{utrs}(\theta)(v) = \phi^{srs}_{utrs}(\theta(v))$ for $v \in \mathcal{V}$
2. $\phi^{srs}_{utrs}(C)[t] = \phi^{srs}_{utrs}(C[\,])[\Box := t]$
3. $\phi^{srs}_{utrs}(\mathcal{R}) = \{\phi^{srs}_{utrs}(l) \to \phi^{srs}_{utrs}(r) \mid l \to r \in \mathcal{R}\}$

本節では型無し項、UTRS、文脈または代入 x について、$\overline{x} = \phi^{srs}_{utrs}(x)$ と略記する。

補題 4.1.3 型無し項 $t \in \mathcal{T}(\Sigma,\mathcal{V})$ および代入 θ について、

$$\overline{t}\,\overline{\theta} = \overline{t\theta}$$

(証明) t の構造帰納法で示す。

基底段階: $t \in \Sigma \cup \mathcal{V}$ について、

1. $t = f \in \Sigma$ のとき、$\overline{f}\,\overline{\theta} = f\overline{\theta} = f = \overline{f\theta}$
2. $t = v \in \mathcal{V}$ のとき、$\overline{v}\,\overline{\theta} = v\overline{\theta} = \overline{v\theta}$

帰納段階: $\overline{t_i}\,\overline{\theta} = \overline{t_i\theta}$ for $i = 0, 1$ と仮定する。このとき、$t = t_0[t_1]$ について、

$$\begin{aligned}
\overline{t}\,\overline{\theta} &= \overline{t_0[t_1]}\,\overline{\theta} \\
&= \overline{t_0}[\overline{t_1}]\,\overline{\theta} && \text{(変換の定義より)} \\
&= \overline{t_0}\,\overline{\theta}[\overline{t_1}\,\overline{\theta}] && \text{(代入の定義より)} \\
&= \overline{t_0\theta}[\overline{t_1\theta}] && \text{(仮定より)} \\
&= \overline{t_0\theta[t_1\theta]} && \text{(変換の定義より)} \\
&= \overline{t_0[t_1]\theta} && \text{(代入の定義より)} \\
&= \overline{t\theta}
\end{aligned}$$

したがって、任意の $t \in \mathcal{T}(\Sigma, \mathcal{V})$ について、$\overline{t}\,\overline{\theta} = \overline{t\theta}$ □

補題 4.1.4 $t \in \mathcal{T}(\Sigma, \mathcal{V})$ について、

$$\overline{C}[\overline{t}] = \overline{C[s]}$$

(証明)

$$\begin{aligned}
\overline{C}[\overline{t}] &= \overline{C[\,]}[\Box := \overline{t}] && \text{(文脈の変換の定義より)} \\
&= \overline{C[\,]}\,\overline{[\Box := t]} && \text{(代入の変換の定義より)} \\
&= \overline{C[\,][\Box := t]} && \text{(補題 4.1.3 より)} \\
&= \overline{C[s]} && \text{(文脈の定義より)}
\end{aligned}$$

□

補題 4.1.5 任意の UTRS $\mathcal{R}$ について、SRS $\overline{\mathcal{R}}$ は $\mathcal{R}$ をシミュレートする。

(証明) 補題 4.1.4、補題 4.1.3 および補題 3.1.4 によって示される。 □

例 4.1.6 例 3.3.6 の $\mathcal{R}'_{map}$ を変換すると、以下のようになる。

$$\overline{\mathcal{R}'_{map}} = \begin{cases} ((map\ f)\ nil) \to nil \\ ((map\ f)\ ((cons\ x)\ xs)) \to ((cons\ (f\ x))\ ((map\ f)\ xs) \end{cases}$$

4.2 UTRS から TRS への変換

定義 4.2.1 $\mathcal{T}(\Sigma, \mathcal{V})$ から $\mathcal{T}_1(\Sigma', \mathcal{V})$ への変換 ϕ^{trs}_{utrs} を、$\phi^{trs}_{utrs}(t) = \phi^{trs}_{srs}(\phi^{srs}_{utrs}(t))$ で定義する。改めて書くと、以下のように表現できる。

1. $\phi^{trs}_{utrs}(v) = v$ if $v \in \mathcal{V}$
2. $\phi^{trs}_{utrs}(f) = f'$ if $f \in \mathcal{V}$
3. $\phi^{trs}_{utrs}(f[t]) = f(\phi^{trs}_{utrs}(t))$ if $f \in \Sigma$ and $f \to r \notin \mathcal{R}$
4. $\phi^{trs}_{utrs}(s[t]) = \circ(\phi^{trs}_{utrs}(s), \phi^{trs}_{utrs}(t))$

補題 4.2.2 任意の UTRS $\mathcal{R}$ について、

$$t \to^{+}_{\mathcal{R}} u \implies \phi^{trs}_{utrs}(t) \to^{+}_{\mathcal{R} \cup \mathcal{A}(\mathcal{R})} \phi^{trs}_{utrs}(u)$$

例 4.2.3 $\mathcal{R}'_{map}$ を変換すると以下のようになる。

$$\phi^{trs}_{utrs}(\mathcal{R}'_{map}) = \begin{cases} \circ(map(f), nil') \to nil' \\ \circ(map(f), \circ(cons(x), xs)) \to \circ(cons(\circ(f, x)), \circ(map(f), xs)) \\ \circ(map', f) \to map(f) \\ \circ(cons', x) \to cons(x) \end{cases}$$

この TRS の停止性は、辞書式経路順序では示せない。2 番目の規則を順序付けるためには、第 1 引数に着目して $map(f) \geq_{lpo} cons(\circ(f, x))$ 即ち $map > cons \wedge map(f) >_{lpo} \circ(f, x)$ 即ち $map > cons \wedge map > \circ$ でなくてはならないが、これでは 3 番目の規則は正しく順序付けられない。

第5章　強計算性依存対法に基づく単純型S式の停止性証明法

本章では、論文 [8] で単純型項書換え系の停止性証明のために導入された強計算依存対法を SRS に導入したい。まず、論文 [8] による単純型を用いて SRS に単純型を導入した単純型 S 式書換え系 (SSRS) を定義する。そして SSRS から STRS への変換を定義し、強計算依存対について議論する。

5.1　単純型項書換え系、単純型S式書換え系

文献 [7] で定義された単純型を使用し、単純型項および単純型 S 式を定義する。

定義 5.1.1 (単純型) 基本型の集合 $\mathcal{B}$ から定義されるすべての単純型の集合 $\mathcal{ST}$ を、演算子 $\times, \to$ によって以下のように帰納的に定義する。

基本型: $\mathcal{B} \in \mathcal{ST}$

関数型: $(\alpha_1 \to \alpha_2) \in \mathcal{ST}$ if $\alpha_1, \alpha_2 \in \mathcal{ST}$

直積型: $(\alpha_1 \times \cdots \times \alpha_n) \in \mathcal{ST}$ if $\alpha_1, \cdots, \alpha_n \in \mathcal{ST}$

ただし、$\times$ は $\to$ より優先度が高く、$\to$ は右結合性を持つとし、冗長な括弧は省略してよいことにする。また、S 式のための単純型の集合 $\mathcal{ST}_{\mathcal{S}} \subseteq \mathcal{ST}$ を、以下の性質を満たす最小の集合とする。

基本型: $\mathcal{B} \in \mathcal{ST}_{\mathcal{S}}$

関数型: $(\alpha_1 \times \alpha_n) \to \alpha \in \mathcal{ST}_{\mathcal{S}}$ if $\alpha_1, \cdots, \alpha_n \in \mathcal{ST}_{\mathcal{S}}$ and $\alpha \in \mathcal{ST}_{\mathcal{S}}$

定義 5.1.2 (単純型項) 特別な構成子 $tp \in \Sigma$ の存在を仮定する。写像 $\tau : \Sigma - \{tp\} \cup \mathcal{V} \to \mathcal{ST}$ を型関数という。これを $\mathcal{T}(\Sigma, \mathcal{V})$ 上に拡張した $\hat{\tau}$ を以下のように定義する。

1. $\hat{\tau}(a) = \tau(a)$ if $a \in \Sigma \cup \mathcal{V}$
2. $\hat{\tau}(a[t_1, \cdots, t_n]) = \alpha$
   if $\hat{\tau}(a) = \alpha_1 \to \cdots \to \alpha_n \to \alpha$ and $\hat{\tau}(t_i) = \alpha_i$ for $i = 1, \cdots, n$
3. $\hat{\tau}(tp[t_1, \cdots, t_n]) = \alpha_1 \times \cdots \times \alpha_n$ if $\hat{\tau}(t_i) = \alpha_i$ for $i = 1, \cdots, n$

以降 $\hat{\tau}$ を単に τ で表す。$t \in \mathcal{T}(\Sigma, \mathcal{V})$ について、$\tau(t)$ を t の型という。$\mathcal{T}(\Sigma, \mathcal{V})$ のうち τ で型が定義されているものを単純型項といい、単純型項の集合を $\mathcal{T}_\tau(\Sigma, \mathcal{V})$ で表す。また、項 $tp[t_1, \cdots, t_n]$ は $(t_1, \cdots, t_n)$ と略記される。

$\tau(t) = \alpha$ であることを明示したいときは、t を t^α と表すことにする。

定義 5.1.3 (単純型項書換え系) 代入及び文脈を型を保存するものに限定し、規則の左辺と右辺の型が等しい規則の集合 $\mathcal{R}$ について、ARS $\langle \mathcal{T}_\tau(\Sigma, \mathcal{V}), \to_\mathcal{R} \rangle$ を単純型項書換え系 (Simply-typed Term Rewriting System; STRS) という。

定義 5.1.4 (単純型 S 式) 特別な基本型 $unit \in \mathcal{B}$ の存在を仮定する。型関数 $\tau : \Sigma \cup \mathcal{V} \to \mathcal{ST}_\mathcal{S} - \{unit\}$ について、これを $\mathcal{S}(\Sigma, \mathcal{V})$ 上に拡張した $\hat{\tau}$ を以下のように定義する。

1. $\hat{\tau}(a) = \tau(a)$ if $a \in \Sigma \cup \mathcal{V}$
2. $\hat{\tau}((t)) = \alpha$ if $\hat{\tau}(t) = unit \to \alpha$
3. $\hat{\tau}((t_0\ t_1\ \cdots\ t_n)) = \alpha$ if $\hat{\tau}(t_0) = \alpha_1 \times \cdots \times \alpha_n \to \alpha$ and $\hat{\tau}(t_i) = \alpha_i$ for $i = 1, \cdots, n$

以降 $\hat{\tau}$ を単に τ で表す。$s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、$\tau(s)$ を s の型という。$\mathcal{S}(\Sigma, \mathcal{V})$ のうち、τ で型が定義されているものを単純型 S 式といい、単純型 S 式の集合を $\mathcal{S}_\tau(\Sigma, \mathcal{V})$ で表す。また、混乱がない場合 $unit$ は省略してよいことにする。

$\tau(s) = \alpha$ であることを明示したいときは、s を s^α と表す。

定義 5.1.5 (単純型 S 式書換え系) 代入及び文脈を型を保存するものに限定し、規則の左辺と右辺の型が等しい規則の集合 $\mathcal{R}$ について、ARS $\langle \mathcal{S}_\tau(\Sigma, \mathcal{V}), \to_\mathcal{R} \rangle$ を単純型 S 式書換え系 (Simply-typed S-expression Rewriting System; SSRS) という。

5.2 SSRS から STRS への変換

まず、SSRS をシミュレートする STRS への変換を定義する。

定義 5.2.1 $\mathcal{S}_\tau(\Sigma, \mathcal{V})$ から $\mathcal{T}_\tau(\Sigma \cup \{\bot\}, \mathcal{V})$ への変換 ϕ^{strs}_{ssrs} を、以下のように定義する。ただし、$\bot \notin \Sigma$ であり、$\tau(\bot) = unit$ とする。

1. $\phi^{strs}_{ssrs}(a) = a$ if $a \in \Sigma \cup \mathcal{V}$
2. $\phi^{strs}_{ssrs}((s)) = \phi^{strs}_{ssrs}(s)[\bot]$
3. $\phi^{strs}_{ssrs}((s_0\ s_1\ \cdots\ s_n)) = \phi^{strs}_{ssrs}(s_0)[(\phi^{strs}_{ssrs}(s_1), \cdots, \phi^{strs}_{ssrs}(s_n))]$

定義 5.2.2 代入 $\theta : \mathcal{V} \to \mathcal{S}_\tau(\Sigma, \mathcal{V})$、文脈 C、規則集合 $\mathcal{R}$ の変換を以下のように定義する。

1. $\phi^{strs}_{ssrs}(\theta)(v) = \phi^{strs}_{ssrs}(\theta(v))$ for $v \in \mathcal{V}$
2. $\phi^{strs}_{ssrs}(C)[t] = \phi^{strs}_{ssrs}(C[\,])[\Box := t]$
3. $\phi^{strs}_{ssrs}(\mathcal{R}) = \{\phi^{strs}_{ssrs}(l) \to \phi^{strs}_{ssrs}(r) \mid l \to r \in \mathcal{R}\}$

単純型 S 式、代入、文脈または SSRS x について、$\overline{x} = \phi^{strs}_{ssrs}(x)$ と略記する。

補題 5.2.3 $s \in \mathcal{S}_\tau(\Sigma, \mathcal{V})$ について、

$$\tau(\overline{s}) = \tau(s)$$

(証明) 式の構造帰納法で示せる。 □

補題 5.2.4 任意の $s \in \mathcal{S}_\tau(\Sigma, \mathcal{V})$ および代入 θ について、

$$\overline{s\theta} = \overline{s}\,\overline{\theta}$$

(証明) 式の構造帰納法で示す。

基底段階: $a \in \Sigma \cup \mathcal{V}$ について、$\overline{a\theta} = a\overline{\theta} = \overline{a}\,\overline{\theta}$

帰納段階: $\overline{s_i\theta} = \overline{s_i}\,\overline{\theta}$ for $i = 0, \cdots n$ と仮定する。このとき $s = (s_0\ s_1\ \cdots\ s_n)$ について、

$$\begin{array}{rcll} \overline{s\theta} & = & \overline{(s_0\theta\ \cdots\ s_n\theta)} & \text{代入の定義より} \\ & = & \overline{s_0\theta}[(\overline{s_1\theta}, \cdots, \overline{s_n\theta})] & \text{変換の定義より} \\ & = & \overline{s_0}\,\overline{\theta}[(\overline{s_1}\,\overline{\theta}, \cdots, \overline{s_n}\,\overline{\theta})] & \text{仮定より} \\ & = & \overline{s_0}[(\overline{s_1}, \cdots, \overline{s_n})]\,\overline{\theta} & \text{代入の定義より} \\ & = & \overline{(s_0\ s_1\ \cdots\ s_n)}\,\overline{\theta} & \text{変換の定義より} \end{array}$$

したがって、任意の $s \in \mathcal{S}_\tau(\Sigma, \mathcal{V})$ について、$\overline{s\theta} = \overline{s}\,\overline{\theta}$ □

補題 5.2.5 $s \in \mathcal{S}(\Sigma, \mathcal{V})$ について、

$$\overline{C}[\overline{s}] = \overline{C[s]}$$

(証明)

$$\begin{array}{rcll} \overline{C}[\overline{s}] & = & \overline{C[\,]}[\Box := \overline{s}] & \text{(文脈の変換の定義より)} \\ & = & \overline{C[\,]}\,\overline{[\Box := s]} & \text{(代入の変換の定義より)} \\ & = & \overline{C[\,][\Box := s]} & \text{(補題 5.2.4 より)} \\ & = & \overline{C[s]} & \text{(文脈の定義より)} \end{array}$$

□

定理 5.2.6 任意の SSRS $\mathcal{R}$ について、STRS $\overline{\mathcal{R}}$ は $\mathcal{R}$ をシミュレートする。

(証明) 補題 5.2.5、補題 5.2.4 および補題 3.1.4 によって示される。 □

補題 5.2.7 すべての $a \in \Sigma \cup \mathcal{V}$ について $\tau(a) \in \mathcal{ST}_\mathcal{S}$ ならば、任意の項 $t \in \mathcal{T}_\tau(\Sigma, \mathcal{V})$ について以下の性質が成り立つ。

$$\tau(t) \in \mathcal{ST}_\mathcal{S} \implies \exists s \in \mathcal{S}_\tau(\Sigma, \mathcal{V}).\ t = \overline{s}$$

(証明) 項の構造に関する帰納法で示す。

基底段階: $t \in \Sigma \cup \mathcal{V}$ のときは明らか。

帰納段階: t_i について $t_i = \overline{s_i}$ となる s_i 存在するとする。ただし $i = 0, \cdots, n$ である。このとき、$t = t_0[(t_1, \cdots, t_n)]$ について、

$$t = \overline{(s_0\ s_1\ \cdots\ s_n)}$$

また、$t' = (t_0, \cdots, t_n)$ は条件を満たさない。

□

補題 5.2.8 すべての $a \in \Sigma \cup \mathcal{V}$ について $\tau(a) \in \mathcal{ST}_{\mathcal{S}}$ ならば、項上の任意の代入 $\theta : \mathcal{V} \to \mathcal{T}_\tau(\Sigma, \mathcal{V})$、$\mathcal{T}_\tau(\Sigma, \mathcal{V})$ 上の文脈 $C[\,]$ について、について以下の性質が成り立つ。

- $\exists \theta' : \mathcal{V} \to \mathcal{S}_\tau(\Sigma, \mathcal{V}).\ \theta = \overline{\theta'}$
- $\tau(C[\,]) \in \mathcal{ST}_{\mathcal{S}} \implies \exists C'[\,].\ C[\,] = \overline{C'[\,]}$

(証明) $\forall v \in \mathcal{V}.\ \tau v \in \mathcal{ST}_{\mathcal{S}}$ なので、$\forall v \in \mathcal{V}.\ \tau(\theta(v)) \in \mathcal{ST}_{\mathcal{S}}$ したがって補題 5.2.7 より、任意の $v \in \mathcal{V}$ について $\overline{s_v} = \theta(v)$ となる $s_v \in \mathcal{S}_\tau(\Sigma, \mathcal{V})$ が存在する。したがって $\theta'(v) = s_v$ は $\overline{\theta'} = \theta$ を満たす。文脈については $\tau(\Box) \in \mathcal{ST}_{\mathcal{S}}$ より明らか。 □

定理 5.2.9 $\mathcal{S}_\tau(\Sigma, \mathcal{V})$ 上の SSRS $\mathcal{R}$ について、以下の性質が成り立つ。

$$\overline{s} \to_{\overline{\mathcal{R}}} \overline{t} \iff s \to_{\mathcal{R}} t$$

(証明) $\Longleftarrow$ は定理 5.2.6 である。$\Longrightarrow$ を示すため、$\overline{s} \to_{\overline{\mathcal{R}}} \overline{t}$ とする。すなわち

$$\exists \theta.\ \exists C[\,].\ \exists l \to r \in \mathcal{R}.\ \overline{s} = C[\overline{l}\theta] \wedge \overline{t} = C[\overline{r}\theta]$$

このとき、補題 5.2.8 より

$$\exists \theta'.\ \exists C'[\,].\ \exists l \to r \in \mathcal{R}.\ \overline{s} = \overline{C'[l\theta']} \wedge \overline{t} = \overline{C'[r\theta']}$$

したがって $s \to_{\mathcal{R}} t$ □

5.3 STRS における SC-DP 法の紹介

文献 [8] で提案された SC-DP による STRS の停止性証明法を紹介する。

定義 5.3.1 STRS $\mathcal{R}$ について、使用される直積型の集合 $upt(\mathcal{R})$ は、以下のように定義される。

$$upt(\mathcal{R}) = \{\alpha_1 \times \cdots \times \alpha_n \mid \exists l \to r \in \mathcal{R}.\, v \in Var(r), \tau(v) = \alpha_1 \times \cdots \times \alpha_n\}$$

$l \to r \in \mathcal{R}$ について、左辺 l の拡張引数 $e\_args(\mathcal{R}, l)$ は、以下のように帰納的に定義される。

1. $args(l) \subseteq e\_args(\mathcal{R}, l)$
2. $u_i \in e\_args(\mathcal{R}, l)$
   if $(\cdots, u_i, \cdots) \in e\_args(\mathcal{R}, l)$ and $\tau(\cdots, u_i, \cdots) \notin upt(\mathcal{R})$

左辺 l の安全な引数の集合 $safe(\mathcal{R}, l)$ とは、以下のように定義される。

$$safe(\mathcal{R}, l) = e\_args(\mathcal{R}, l) \cup \{u \in Sub(l) \mid u \neq l, \tau(u) \in \mathcal{B} \cup upt(\mathcal{R})\}$$

定義 5.3.2 (強計算性) 以下の条件を満たすとき、項 t は STRS $\mathcal{R}$ において強計算性を持つといい、$SC(\mathcal{R}, t)$ と表す。

- $\tau(t) \in \mathcal{B} \cup upt(\mathcal{R})$ and $SN(\mathcal{R}, t)$ または、
- $\tau(t) = \alpha_1 \times \cdots \times \alpha_n \notin upt(\mathcal{R}), SN(\mathcal{R}, t)$ and
  $\forall t_1, \cdots, t_n.\, (t \to_{\mathcal{R}}^* (t_1, \cdots, t_n) \implies SC(\mathcal{R}, t_i))$ または、
- $\tau(t) = \alpha \to \beta$ and $\forall u^{\alpha}.\, (SC(\mathcal{R}, u) \implies SC(\mathcal{R}, t[u]))$

強計算性は停止性の十分条件であることが示されている。すなわち $SC(\mathcal{R}, t) \implies SN(\mathcal{R}, t)$ が成立する。

定義 5.3.3 (PFP) STRS $\mathcal{R}$ について、任意の葉文脈 $C[\,]$、規則 $l \to C[t] \in \mathcal{R}$ に対して、根文脈 $S[\,]$ および $s \in safe(\mathcal{R}, l) \cup \Sigma$ が存在して、$t = S[s]$ となるとき、$\mathcal{R}$ は Plain Function Passing; PFP である、という。

定義 5.3.4 (印付項) すべての $f \in \mathcal{D}_{\mathcal{R}}$ について、記号 $f^{\sharp}$ を用意する。また、$t = a[t_1, \cdots, t_n] \in \mathcal{T}(\Sigma, \mathcal{V})$ について、印付項 $t^{\sharp}$ を以下のように定義する。

- $t^\sharp = a^\sharp[t_1, \cdots, t_n]$ if $a \in \mathcal{D}_\mathcal{R}$
- $t^\sharp = t$ otherwise

定義 5.3.5 (強計算依存対) $S[\,], S'[\,]$ を新しい変数から構成される、関数型でない根文脈とする。葉文脈 $C[\,]$ および $l \to C[t] \in \mathcal{R}$ について、以下の性質を満たすとき $\langle S[l^\sharp], S'[t^\sharp]\rangle \in DP_{SC}(\mathcal{R})$ と定義する。

- $root(t) \in \mathcal{D}_\mathcal{R}$ かつ
- 任意の根文脈 $S''[\,]$ および $s \in safe(\mathcal{R}, l)$ について、$t \neq S''[s]$

定義 5.3.6 (依存鎖) 依存対の集合 $C \subseteq DP_{SC}(\mathcal{R})$ について、以下の条件を満たす列 $\langle u_0^\sharp, v_0^\sharp\rangle\langle u_1^\sharp, v_1^\sharp\rangle \cdots \in C^*$ の集合を $\langle C, \to_\mathcal{R}\rangle$-*chain* という。すなわち代入の列 $\theta_0, \theta_1, \cdots$ が存在して、すべての i について

- $\forall t \in args(u_i\theta_i) \cup args(v_i\theta_i).\ SC(\mathcal{R}, t)$ かつ
- $v_i\theta_i \to_\mathcal{R}^* u_{i+1}\theta_{i+1}$

また、$\langle DP_{SC}(\mathcal{R}), \to_\mathcal{R}\rangle$-*chain* を $DP_{SC}(\mathcal{R})$-*chain* で表す。

命題 5.3.7 PFP-STRS $\mathcal{R}$ について、無限の $DP_{SC}(\mathcal{R})$-*chain* が存在しなければ、$SN(\mathcal{R})$ である。

定義 5.3.8 強計算依存グラフ $DG_{SC}(\mathcal{R})$ とは、$DP_{SC}(\mathcal{R})$ を頂点とし、すべての $ee' \in DP_{SC}(\mathcal{R})$-*chain* を辺とした有向グラフである。$DG_{SC}(\mathcal{R})$ の強連結成分を強計算再帰成分といい、すべての強計算再帰成分の集合を $RC_{SC}(\mathcal{R})$ で表す。

命題 5.3.9 すべての $C \in RC_{SC}(\mathcal{R})$ について無限の $\langle C, \to_\mathcal{R}\rangle$-*chain* が存在しなければ、$SN(\mathcal{R})$ である。

5.4 SC-DP 法の SSRS への適用

本節では、SC-DP 法を SSRS に適用するために必要となる定義を与える。

定義 5.4.1 (S式の引数、拡張引数) S式sにおける引数の集合$args(s)$を以下のように定義する。

$$args((s_0\ s_1\ \cdots\ s_n)) = \{s_1, \cdots, s_n\}$$

また拡張引数の集合$s_args(s)$を、以下の性質を満たす最小の集合として定義する。

1. $a \in s\_args(a)$ if $a \in \Sigma \cup \mathcal{V}$
2. $s\_args(s_0) \cup args(s) \subseteq s\_args((s_0\ s_1\ \cdots\ s_n))$

また、安全な引数の集合$s_safe(l)$を、以下のように定義する。

$$s\_safe(l) = s\_args(l) \cup \{s \in Sub(l) \mid s \neq l, \tau(s) \in \mathcal{B}\}$$

補題 5.4.2 SSRS $\mathcal{R}$を変換したSTRS $\overline{\mathcal{R}}$について、

$$upt(\overline{\mathcal{R}}) = \emptyset \tag{5.1}$$
$$e\_args(\overline{\mathcal{R}}, \overline{l}) = \overline{s\_args(l)} \tag{5.2}$$
$$safe(\overline{\mathcal{R}}, \overline{l}) = \overline{s\_safe(l)} \tag{5.3}$$

(証明) 直積型を持つ単純型S式は存在しないので、(5.1)は明らか。これから(5.2)、(5.3)も導かれる。 □

定義 5.4.3 (PFP-SSRS) 任意の葉文脈$C[\,]$および規則$l \to C[s] \in \mathcal{R}$に対し、$root(s) \in \mathcal{V}$ならば、根文脈$S[\,]$およびS式$t \in s_safe(l)$が存在して$s = S[t]$となるとき、SSRS $\mathcal{R}$は Plain Function Passing; PFP である、という。

補題 5.4.4 SSRS $\mathcal{R}$がPFPならば変換後のSTRS $\overline{\mathcal{R}}$はPFPである。

(証明) 定義5.4.3に補題5.4.2を適用すれば定義5.3.3と等価になることから示される。 □

定義 5.4.5 (印付S式) すべての$f \in \mathcal{D}_{\mathcal{R}}$について、印付記号$f^\sharp$を用意する。また、$s \in \mathcal{S}(\Sigma, \mathcal{V})$について、印付S式$s^\sharp$を以下のよう帰納的に定義する。

- $f^\sharp = f$ if $f \in \mathcal{C}_{\mathcal{R}} \cup \mathcal{V}$

- $(s_0\ s_1\ \cdots\ s_n)^\sharp = (s_0^\sharp\ s_1\ \cdots\ s_n)$

補題 5.4.6 $s \in \mathcal{S}_\tau(\Sigma, \mathcal{V})$ について、$\overline{s^\sharp} = \overline{s}^\sharp$

(証明) 式の構造に関する帰納法で示せる。 □

定義 5.4.7 (S 式における強計算依存対) $S[\,], S'[\,]$ を新しい変数から構成され、基本型を持つ根文脈とする。SSRS $\mathcal{R}$ について、以下の性質を満たす印付 S 式の対 $\langle S[l^\sharp], S'[t^\sharp]\rangle$ を強計算依存対といい、すべての強計算依存対の集合を $DP_{SC}(\mathcal{R})$ で表す。すなわち葉文脈 $C[\,]$ および $l \to C[t] \in \mathcal{R}$ が存在して

- $root(t) \in \mathcal{D}_\mathcal{R}$ かつ
- 任意の根文脈 $S''[\,]$ および $s \in s\_safe(l)$ について、$t \neq S''[s]$

補題 5.4.8 依存対と変換後の依存対について、以下の性質が成り立つ。

$$\langle u^\sharp, v^\sharp\rangle \in DP_{SC}(\mathcal{R}) \iff \langle \overline{u}^\sharp, \overline{v}^\sharp\rangle \in DP_{SC}(\overline{\mathcal{R}})$$

(証明) 補題 5.4.2 および補題 5.4.6 より明らか。 □

定義 5.4.9 (依存鎖) 依存対の集合 $C \subseteq DP_{SC}(\mathcal{R})$ について以下の条件を満たす列 $\langle u_0^\sharp, v_0^\sharp\rangle\langle u_1^\sharp, v_1^\sharp\rangle \cdots \in C^*$ の集合を $\langle C, \to_\mathcal{R}\rangle$-*chain* という。すなわち代入の列 $\theta_0, \theta_1, \cdots$ が存在して、すべての i について

- $\forall s \in s\_args(u_i\theta_i) \cup s\_args(v_i\theta_i).\ SC(\overline{\mathcal{R}}, \overline{s})$ かつ
- $v_i\theta_i \to_\mathcal{R}^* u_{i+1}\theta_{i+1}$

また、$\langle DP_{SC}(\mathcal{R}), \to_\mathcal{R}\rangle$-*chain* を $DP_{SC}(\mathcal{R})$-*chain* で表す。

依存対の集合 $C \subseteq DP_{SC}(\mathcal{R})$ について、変換後の依存対の集合 $\overline{C}$ を、以下のように定義する。

$$\overline{C} = \{\langle \overline{u}^\sharp, \overline{v}^\sharp\rangle \mid \langle u^\sharp, v^\sharp\rangle \in C\}$$

補題 5.4.10 依存対の集合 $C \subseteq DP_{SC}(\mathcal{R})$ について、

$$\langle u_0^\sharp, v_0^\sharp\rangle\langle u_1^\sharp, v_1^\sharp\rangle \cdots \in \langle C, \to_{\mathcal{R}}\rangle\text{-}chain \implies \langle \overline{u_0}^\sharp, \overline{v_0}^\sharp\rangle\langle \overline{u_1}^\sharp, \overline{v_1}^\sharp\rangle \cdots \in \langle C, \to_{\mathcal{R}}\rangle\text{-}chain$$

(証明) 定理 5.2.9 により明らか。 □

定義 5.4.11 強計算依存グラフ $DG_{SC}(\mathcal{R})$ とは、$DP_{SC}(\mathcal{R})$ を頂点とし、すべての $ee' \in DP_{SC}(\mathcal{R})\text{-}chain$ を辺とした有向グラフである。$DG_{SC}(\mathcal{R})$ の強連結成分を強計算再帰成分といい、すべての強計算再帰成分の集合を $RC_{SC}(\mathcal{R})$ で表す。

定理 5.4.12 すべての $C \in RC_{SC}(\mathcal{R})$ について無限の $\langle C, \to_{\mathcal{R}}\rangle\text{-}chain$ が存在しなければ、$SN(\mathcal{R})$ である。

(証明) 補題 5.4.10 および命題 5.3.9 より明らか。 □

5.5　部分項基準の SSRS への適用

1 階の TRS について論文 [4] で提案され、論文 [8] で拡張、STRS 上に導入された部分項基準による停止性証明法を SSRS 上に導入する。

定義 5.5.1 (部分式) $p \in S\_Pos(s)$ について、s の位置 p での部分式 $s|_p$ を以下のように帰納的に定義する。

1. $s|_\varepsilon = s$
2. $(s_0\ \cdots\ s_n)|_{ip} = s_i|_p$ for $i = 1, \cdots, n$ and $p \in S\_Pos(s_i)$

また、S 式 s, t について、部分式関係 $\geq_{ssub}$ および $>_{ssub}$ を、以下のように定義する。

$$s \geq_{ssub} t \quad \stackrel{def}{\iff} \quad \exists p \in S\_Pos(s).\ s|_p = t$$
$$s >_{ssub} t \quad \stackrel{def}{\iff} \quad s \geq_{sub} t \wedge s \neq t$$

定義 5.5.2 (SSRS における部分項基準) SSRS $\mathcal{R}$ および依存対の部分集合 $C \subseteq DP_{SC}(\mathcal{R})$ について、写像 $\pi : \mathcal{D}_{\mathcal{R}} \to \mathbb{N}^*$ が存在して以下の条件を満たすとき、C は部分項基準を満たすという。

(a) $\exists\langle u^\sharp, v^\sharp\rangle \in C.\ u|_{\pi(root(u))} >_{ssub} v|_{\pi(root(v))}$ かつ、

(b) $\forall\langle u^\sharp, v^\sharp\rangle \in C.$

- $u|_{\pi(root(u))} \geq_{ssub} v|_{\pi(root(v))}$,
- $\forall q < \pi(root(v)).\ (q \neq \varepsilon \implies root(v|_q) \in \mathcal{C}_\mathcal{R})$

補題 5.5.3 $C \subseteq DP_{SC}(\mathcal{R})$ が部分項基準を満たすなら、C のすべての要素を無限に含む $\langle C, \to_\mathcal{R}\rangle$-*chain* は存在しない。

(証明) 無限 $\langle C, \to_\mathcal{R}\rangle$-*chain* $\langle u_0^\sharp, v_0^\sharp\rangle\langle u_1^\sharp, v_1^\sharp\rangle\cdots$ の存在を仮定する。ここで $\theta_0, \theta_1, \cdots$ を、すべての i について

$$\forall s \in s\_args(v_i\theta_i).\ SC(\overline{\mathcal{R}}, \overline{s}) \wedge v_i^\sharp\theta_i \to_\mathcal{R}^* u_{i+1}^\sharp\theta_{i+1}$$

となる代入の列とする。強計算性の性質より、すべての t について

$$SC(\overline{\mathcal{R}}, t) \implies SN(\overline{\mathcal{R}}, t)$$

なので、任意の $s \in s\_args(v_i)$ について $SN(\overline{\mathcal{R}}, \overline{s\theta_i})$ が成立し、定理 5.2.6 より $SN(\mathcal{R}, s\theta_i)$ が成立する。

ここで $p_0, p_1, \cdots$ を、$p_i = \pi(root(u_i))$ となる位置の列とする。このとき $v_i^\sharp\theta_i \to_\mathcal{R}^* u_{i+1}^\sharp\theta_{i+1}$ から $root(v_i) = root(u_{i+1})$ したがって $\pi(v_i) = p_{i+1}$ となる。したがって部分項基準 (b) より、無限列

$$u_0\theta_0|_{p_0} \geq_{ssub} v_0\theta_0|_{p_1} \to_\mathcal{R}^* u_1\theta_1|_{p_1} \geq_{ssub} v_1\theta_1|_{p_2} \to_\mathcal{R}^* \cdots$$

が存在する。部分項基準 (a) よりこの列は無限の $>_{ssub}$ 関係を含むが、$(>_{ssub} \cdot \to_\mathcal{R}^*) \subseteq (\to_\mathcal{R}^* \cdot >_{ssub})$ が成立し、$>_{ssub}$ は整礎性を持つので、$u_0\theta_0|_{p_0}$ から始まる無限の書換え系列を得ることができる。$p_0 > \varepsilon$ よりこれは $\forall s \in s\_args(u_i).\ SN(\mathcal{R}, s)$ に矛盾する。 □

定理 5.5.4 PFP-SSRS $\mathcal{R}$ について、すべての再帰成分 $C \in RC_{SC}(\mathcal{R})$ が部分項基準を満たすとき、$SN(\mathcal{R})$ である。

(証明) 定理 5.4.12 および、補題 5.5.3 による。 □

5.6　証明例

部分項基準を用いた SCDP 法を使用して、いくつかの高階関数の停止性証明例を示す。

例 5.6.1 (型付き foldl) $\mathcal{R}_{tfoldl}$ は左畳み込み関数を実現する SSRS である。

$$\mathcal{R}_{tfoldl} = \begin{cases} (Foldl^{(N\times N\to N)\times N\times L\to N}\ f\ x\ nil) \to x \\ (Foldl\ f\ x\ (cons^{N\times L\to L}\ y\ ys)) \to (Foldl\ f\ (f\ x\ y)\ ys) \end{cases}$$

この SSRS の再帰成分は以下のようになる。

$$RC_{SC}(\mathcal{R}_{tfoldl}) = \left\{ \{\langle (Foldl^\sharp\ f\ x\ \underline{(cons\ y\ ys)}), (Foldl^\sharp\ f\ (f\ x\ y)\ \underline{ys})\rangle\} \right.$$

下線部に注目し、$\pi(Foldl) = 3$ とすれば部分項基準を満たすので、補題 5.5.4 により、$\mathcal{R}$ の停止性が証明される。

例 5.6.2 (型付き map) $\mathcal{R}_{tmap}$ は map 関数を実現する SSRS である。

$$\mathcal{R}_{tmap} = \begin{cases} (Map^{(N\to N)\times L\to L}\ f\ nil) \to nil \\ (Map\ f\ (cons^{N\times N\to L}\ x\ xs)) \to (cons\ (f\ x)\ (Map\ f\ xs)) \end{cases}$$

この再帰成分は以下のようになる。

$$RC_{SC}(\mathcal{R}_{tmap}) = \left\{ \{\langle (Map^\sharp\ f\ \underline{(cons\ x\ xs)}), (Map^\sharp\ f\ \underline{xs})\rangle\} \right.$$

例 5.6.1 と同様、$\pi(Map) = 2$ とすれば停止性が示される。

例 5.6.3 SSRS $\mathcal{R}_{tfoldl} \cup \mathcal{R}_{tmap}$ の停止性は、

$$RC_{SC}(\mathcal{R}_{tfoldl} \cup \mathcal{R}_{tmap}) = RC_{SC}(\mathcal{R}_{tfoldl}) \cup RC_{SC}(\mathcal{R}_{tmap})$$

なので、例 5.6.1 および例 5.6.2 と同様に示される。

第6章　まとめと今後の課題

本研究では、高階書換え系の変換による停止性証明法を研究した。まず一般に停止性証明に使える書換え系の変換の性質を示した。

特にSRSについて、1階のTRSに変換する方法を提案し、停止性証明に使用できることを示した。ただし入力となるSRSに、規則の左辺の部分項の根記号が変数でない、という制限が必要である。この制限はプログラミング言語の表現力に大きな影響を与えるわけではないと思うが、この制限をはずす変換も考えている。

定義 6.0.1 $\mathcal{A}(\mathcal{R})$ の逆のような操作を行うTRS $\mathcal{U}(\mathcal{R})$ を、以下のように定義する。逆といっても異なる記号を使うので、停止性を保存する。

$$\mathcal{U}(\mathcal{R}) = \{f(v_1, \cdots, v_n) \to \bullet(f', v_1, \cdots, v_n) \mid f \in \Sigma\}$$

また、$\circ$ 記号を直接 $\bullet$ 記号に書き換えるTRS $\mathcal{B}$ を定義する。

$$\mathcal{B} = \{\circ(v_0, \cdots, v_n) \to \bullet(v_0, \cdots, v_n) \mid n \in \mathbb{N}\}$$

さらに以下のような変換 $\beta \subseteq \to_{\mathcal{B}}^*$ を定義する。

1. $\beta(a) = a$ if $a \in \Sigma \cup \mathcal{V}$
2. $\beta(\circ(v, t_1, \cdots, t_n)) = \bullet(v, \beta(t_1), \cdots, \beta(t_n))$ if $v \in \mathcal{V}$
3. $\beta(f(t_1, \cdots, t_n)) = f(\beta(t_1), \cdots, \beta(t_n))$ otherwise

予想 6.0.2 $\phi_{srs}^{trs}(\mathcal{R})' = \{\beta(\phi_{srs}^{trs}(l)) \to \phi_{srs}^{trs}(r) \mid l \to r \in \mathcal{R}\}$ と定義する。このとき、任意のSRS $\mathcal{R}$ について、TRS $(\phi_{srs}^{trs}(\mathcal{R})' \cup \mathcal{A}(\mathcal{R}) \cup \mathcal{U}(\mathcal{R}) \cup \mathcal{B})$ は $\mathcal{R}$ をシミュレートする。

また、SSRSからSTRSへの変換を定義し、SSRSの停止性をSTRSの停止性に帰着して証明できることを示した。これを元に、STRS上で研究された強計算性依存対法をSSRSに導入し、この方法が適用できる条件であるPFP-STRSの性

質を、SSRS 上で再定義した。また、判定に使われる部分項基準を SSRS 上で再定義することで、変換を行わずに停止性証明を行うことができるようになった。

SCDP 法は、分割統治法が効果的に働く。2 つの STRS $\mathcal{R}_1, \mathcal{R}_2$ について、

$$RC_{SC}(\mathcal{R}_1 \cup \mathcal{R}_2) = RC_{SC}(\mathcal{R}_1) \cup RC_{SC}(\mathcal{R}_2) \tag{6.1}$$

が成立する場合が多いからである。特に SSRS においては、変換後の STRS の $safe(\overline{\mathcal{R}}, l) = \overline{s\_safe(l)}$ が R に依存しないため、簡単な条件で式 6.1 が成立することが期待できる。この条件を示すことは今後の課題としたい。

謝辞

本研究を行うにあたり、貴重なお時間を割いて丁寧にご指導くださいました坂部俊樹教授、酒井正彦教授、草刈圭一朗講師、西田直樹助手に謹んで感謝いたします。また、本研究をお助けくださいました坂部・酒井研究室の皆様に感謝いたします。

参考文献

[1] Tyrolean Termination Tool web page. `http://cl2-informatik.uibk.ac.at/ttt/`.

[2] AProVE web page. `http://aprove.informatik.rwth-aachen.de/index.asp?subform=home.html`.

[3] C*i*ME web page. `http://cime.lri.fr/`.

[4] Dependency pairs revisited. In *Proc. of the 15th Int. Conf. on Rewriting Techniques and Applications, LNCS 3091 (RTA04)*, pp. 249–268.

[5] Takahito Aoto and Toshiyuki Yamada. Termination of simply typed term reqriting by translation and labelling. In *RTA'03 Lecture Notes in Computer Science 2706*, pp. 380–394, 2003.

[6] F.Baader and T.Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[7] K.Kusakari. On proving termination of term rewriting systems with higher-order variables. In *IPSJ*, pp. 35–45, 2001.

[8] K.Kusakari and M.Sakai. Enhancing dependency pair method by strong computability in simply-typed term rewriting. In *Applicable Algebra in Engineering, Communication and Computin manuscript*, 2006.

[9] N.Dershowitz. Termination of rewriting. In *J.Symbolic Computation*, Vol. 3, pp. 69–116, 1987.

[10] 桜井啓大. 引数減少原理に基づいた単純型項書換え系の停止性証明. 情報工学コース卒業研究報告書, 2003.

[11] Yoshihito Toyama. Termination of s-expression rewriting systems: Lexicographic path ordering for higher-order terms. In Vincent van Oostrom, editor, *RTA*, Vol. 3091 of *Lecture Notes in Computer Science*, pp. 40–54. Springer, 2004.