LA-001

関数プログラムの再帰構造解析と強計算性に基づく十分完全性の証明法

Proving Sufficient Completeness of Functional Programs
based on Recursive Structure Analysis and Strong Computability

櫻井敬大†, 草刈圭一朗‡, 西田直樹‡, 酒井正彦‡, 坂部俊樹‡

概要: 十分完全性を持つ関数プログラムは任意の入力値に対して出力値の存在が保証されるため，十分完全性の証明法の研究は重要である．本論文では，依存対法と呼ばれる停止性証明法を土台にした再帰構造の静的解析に基づく十分完全性の証明法を与える．本手法はプログラマーの直感に合致し，それゆえに適用範囲も広く強力な手法である．また，提案した手法の正当性を，型付き λ 計算の停止性証明で導入された強計算性の概念を用いて示す．本手法は，依存対法に強計算性を組み合わせる事に成功した初めての成果である．

1. はじめに

関数プログラミング言語は，計算の対象を関数定義の形で宣言的に記述することによりプログラミングを行う言語である．関数プログラミング言語の特長としては参照透明性や高階関数の導入による抽象化などが挙げられる．関数プログラミング言語における重要な性質の一つに十分完全性と呼ばれる概念がある．関数プログラムが十分完全性を持つとは，任意の入力値に対して出力値が存在することである．

関数プログラムに操作的意味を与える計算モデルの1つに項書換え系と呼ばれるモデルがある [2]．例えば，自然数上の加算と乗算に対応する項書換え系 R_1, R_2 は以下のように与えられる．なお，慣例に従い，自然数 $0,1,2,\ldots$ を項 $0, s(0), s(s(0)),\ldots$ で表している．

$$R_1 = \left\{ \begin{array}{rcl} +(x,0) & \to & x \\ +(x,s(y)) & \to & s(+(x,y)) \end{array} \right.$$

$$R_2 = R_1 \cup \left\{ \begin{array}{rcl} \times(x,0) & \to & 0 \\ \times(x,s(y)) & \to & +(\times(x,y),x) \end{array} \right.$$

項書換え系は関数プログラムで広く利用されている高階関数を直接取り扱うことができないため，高階関数を直接取り扱える高階書き換え系の研究が近年盛んに行われている．しかし，高階書換え系の研究は理論的興味が優先されてきたため，関数プログラミング言語のモデルとして見た場合には，不必要に高度な表現力を持つ定式化が行われてきた．この反省にもとづき我々は，理論的に取り扱いやすいように制限しながらも関数型プログラムに操作的意味を与えるのに十分な表現力を持つ高階の書き換え系である単純型項書換え系 (simply-typed term rewriting system; STRS) を提案した [3]．例えば代表的な高階関数である (左) 畳み込み関数 $foldl$ は次の STRS R_3 で与えることができる．

$$\left\{ \begin{array}{rcl} foldl(f,z,nil) & \to & z \\ foldl(f,z,cons(x,xs)) & \to & foldl(f,f(z,x),xs) \end{array} \right.$$

ここで，慣例に従い構成子 $nil, cons$ を用いてリストを表現している．関数 $foldl$ を用いると，リスト $([x_1,\ldots,x_n])$ の要素の総和 $(x_1+\cdots+x_n)$ を与える関数 sum と総積 $(x_1\times\cdots\times x_n)$ を与える関数 $prod$ を次の STRS R_4, R_5 のように与えることができる．

$$R_4 = R_1 \cup R_3 \cup \{sum(xs) \to foldl(+,0,xs)\}$$
$$R_5 = R_2 \cup R_3 \cup \{prod(xs) \to foldl(\times,s(0),xs)\}$$

この例のように高階関数を用いると高いレベルの抽象化を実現でき，言語の表現力やプログラムの再利用性を高めることができる．

ところで，上述の関数 sum や $prod$ は十分完全性を持つ (任意の入力値に対し出力値が存在する) だろうか？この事実は直感的に正しいと考えられるが，機械的に検証するのはかなり難しい．この困難は高階変数の理論的取り扱いの難しさに起因する．

例として，STRS R_3 を考えてみる．多くのプログラマーは $foldl$ の定義は十分完全性を持つと考えるであろうが，その理由は再帰定義の部分で第3引数 (次式の下線部) が減少し，それゆえに有限回の簡約で出力値が求まると考えるためであろう．

$$foldl(f,z,\underline{cons(x,xs)}) \to foldl(f,f(z,x),\underline{xs})$$

しかしながら，次の関数 foo を R_3 に追加すると状況が一変し，直感どおりの動作をしなくなる．

$$R_6 = R_3 \cup \{foo(z,x) \to foldl(foo,z,cons(x,nil))\}$$

実際，実行時において $foldl$ と foo は相互に再帰呼び出しを繰り返し続け，出力値は存在しない．すなわち，実行時には次式の下線部の部分にも (相互) 再帰関係が発生するため，再帰構造の動的解析を行う場合は高階変数 f を無視できないのである．

$$\underline{foldl(f,z,cons(x,xs))} \to foldl(f,\underline{f(z,x)},xs)$$

もちろん，高階変数 f に代入されうる全ての関数を解析すれば，R_4, R_5 が十分完全性を持ち R_6 が持たないことを示すことができる．しかしながら，プログラミング時に高階変数 f に代入される可能性がある全ての関数を考慮に入れるプログラマーは皆無であろう．それにも関わらず，多くのプログラマーは $foldl$ の定義が妥当であると確信するであろう．しかし前述の議論より，この直感が正当であるためには何らかの制限が必要であることが見て取れる．本研究では，この直感に従った十分完全性の証明法，すなわち，

(I) 高階変数を考慮せず再帰構造の静的解析のみによる十分完全性証明法，

を提案する．また，この証明法が妥当であるための十分条件を調査する．結論として，以下の4条件を全て満たすことが (I) が健全であるための十分条件となることを述べる．

(II) 擬簡約性

(III) 高階成長性

名古屋大学大学院情報科学研究科
†tsakurai@sakabe.i.is.nagoya-u.ac.jp
‡{kusakari,nishida,sakai,sakabe}@is.nagoya-u.ac.jp

(IV) 再帰構造が基本型限定
(V) 比較引数が非増加的

なお，先程の foo の例では条件 (IV) が成立しないため再帰構造の静的解析が適用できない．これらの 4 条件は，foo の例のような人工的な例を除く多くの関数プログラムが持つ性質であり，それゆえに提案した手法は広い範囲のプログラムに対して有効である．なお，この健全性の証明は Tait が型付き λ 計算の停止性証明に導入した強計算性の概念を用いている [5]．なお本論文の成果は，依存対法に基づいている．これは Arts と Giesl が項書換え上で提案し [1] STRS 上へは草刈が拡張した [3]．依存対法に強計算性の概念を導入することに成功したのは本論文が初めてである．

2. 準備

本節では文献 [3] に基づき，論文中で必要となる単純型項書換え系に関する諸概念を与える．

関数記号の集合 Σ と変数記号の集合 $\mathcal{V}$ から生成される**項**の全体からなる集合 $\mathcal{T}(\Sigma,\mathcal{V})$ は次のように帰納的に定義される; $a\in\Sigma\cup\mathcal{V}$ かつ $t_1,\ldots,t_n\in\mathcal{T}(\Sigma,\mathcal{V})$ ならば $a(t_1,\ldots,t_n)\in\mathcal{T}(\Sigma,\mathcal{V})$．項 $a()$ は単に a と記す．2 つの項 s と t が構文的に等しいことを $s\equiv t$ で表す．また，$s\equiv a(s_1,\ldots,s_n)$ とするとき，$s(t_1,\ldots,t_m)$ と書いて項 $a(s_1,\ldots,s_n,t_1,\ldots,t_m)$ を表す．項 t に出現する全ての変数の集合を $Var(t)$ で表す．項 $t\equiv a(t_1,\ldots,t_n)$ の根の位置の記号 a を $root(t)$ で記す．

代入は変数から項への関数である．代入 θ の項上への拡張，すなわち $\theta(a(t_1,\ldots,t_n))$ は，$a\in\Sigma$ のときには $a(\theta(t_1),\ldots,\theta(t_n))$ で，$a\in\mathcal{V}$ かつ $\theta(a)=a'(t'_1,\ldots,t'_k)$ のときには $a'(t'_1,\ldots,t'_k,\theta(t_1),\ldots,\theta(t_n))$ で定義される．$\theta(t)$ を $t\theta$ で略記する．**文脈**とは穴と呼ばれる特別な関数記号 $\Box$ が，葉の位置，すなわち，引数を持たない形式で一箇所だけ出現する項である．文脈 $C[\,]$ 中の $\Box$ を項 t で置き換えることによって得られる項を $C[t]$ で表す．項 t' が項 t の**部分項**であるとは，ある文脈 $C[\,]$ が存在して $t\equiv C[t']$ となることである．項 t の部分項全体を $Sub(t)$ で記す．

空でない**基本型**の集合を $\mathcal{B}$ で表す．$\mathcal{B}$ から生成される**単純型**の集合 $\mathcal{S}$ は，右結合性を持つ型構成子 $\to$ を用いて $\mathcal{S}::=\mathcal{B}\mid(\mathcal{S}\to\mathcal{S})$ として定義される．本論文では，単純型を単に型と呼ぶ事がある．**型関数** τ は $\Sigma\cup\mathcal{V}$ から $\mathcal{S}$ への関数である．項 $t\equiv a(t_1,\ldots,t_n)$ が単純型 α を持つとは，$\tau(a)=\alpha_1\to\cdots\to\alpha_n\to\alpha$ かつ，各 t_i が単純型 α_i を持つことである．単純型を持つ項を**単純型項**と呼ぶ．全ての単純型項からなる集合，全ての基本型を持つ項からなる集合，全ての変数を含まない単純型項からなる集合を，それぞれ $\mathcal{T}_\tau(\Sigma,\mathcal{V}),\mathcal{T}^{\mathcal{B}}_\tau(\Sigma,\mathcal{V}),\mathcal{T}_\tau(\Sigma)$ で記す．項 t の基本型を持つ部分項の全体を $Sub_{\mathcal{B}}(t)$ で記す．$\mathcal{V}_h$ で高階変数，すなわち，基本型でない単純型を持つ変数全てからなる集合を記す．なお本論文中では，代入や文脈は型の整合性を崩さないもののみを扱う．すなわち，単純型項 t に対し $t\theta$ や $C[t]$ のように書いた場合には，$t\theta, C[t]$ は共に単純型を持つとする．

単純型書き換え規則とは $root(l)\in\Sigma$, $Var(l)\supseteq Var(r)$, $\tau(l)=\tau(r)\in\mathcal{B}$ の条件を満たす単純型項 l,r の対 (l,r) であり，$l\to r$ と記す．**単純型項書換え系** (simply-typed term rewriting system; STRS) とは単純型書き換え規則の有限集合である．STRS R において単純型項 s が t に書き換えられるとは，ある規則 $l\to r\in R$ と代入 θ と文脈 $C[\,]$ が存在して $s\equiv C[l\theta]$ かつ $t\equiv C[r\theta]$ となることである．このとき，$s\to_R t$，または単に $s\to t$ と記す．

R を STRS とする．R の規則 $l\to r$ の左辺 l の根の位置に出現する関数記号 $(root(l))$ を**被定義記号**と呼び，R の被定義記号全体からなる集合を $\mathcal{D}_R$ と記す．また，R の被定義記号でない関数記号を R の**構成子**と呼び，R の構成子全体からなる集合を $\mathcal{C}_R$ で記す．単純型項 t が**値**であるとは $\forall t'\in Sub_{\mathcal{B}}(t), root(t')\in\mathcal{C}_R$ となることである．R における値の集合を $Val(R)$ で記す．書換え可能な項が複数存在する場合，直下の部分項が全て値である項を書き換える戦略を**値呼び** (call by value) と呼ぶ．値呼び戦略による書換えを $\to_{cbv}$ で記す．また，任意の $x\in Var(t)$ に対し $\theta(x)\in Val(R)$ となる代入 θ を t に対する**値代入**と呼ぶ．

R を STRS とする．R が**停止性** (strongly normalizing) を持つとは $(SN(R)$ と記す$)$，$t_0\to_R t_1\to_R\cdots$ のような無限列が無いことである．項 t が**十分完全性** (sufficient completeness) を持つとは $(SC_{cbv}(t)$ と記す$)$，ある $c\in Val(R)$ について $t\to^*_{cbv}c$ を満たすことである．R が十分完全性を持つとは $(SC_{cbv}(R)$ と記す$)$，任意の $t\in\mathcal{T}_\tau(\Sigma)$ に対し $SC_{cbv}(t)$ となることである (本論文では値呼び戦略に特化して議論する)．R が**擬簡約化可能** (quasi-reducible) であるとは $(QR(R)$ と記す$)$，任意の $f\in\mathcal{D}_R$ と $c_1,\ldots,c_n\in Val(R)$ に対し，$\tau(f(c_1,\ldots,c_n))\in\mathcal{B}$ ならば $f(c_1,\ldots,c_n)$ が書き換え可能であることである．

各関数記号 $f\in\Sigma$ に対し，**印付記号** $f^\#$ を用意する．また，$t^\#$ で t の root 記号を対応する印付記号で置き換えた項を表す．ここで，$root(t)\in\mathcal{V}$ の場合には $t^\#\equiv t$ とする．$u\to r\in R$ かつ $v\in Sub_{\mathcal{B}}(r)$ かつ $root(v)\in\mathcal{D}_R\cup V_h$ であるとき，項の対 $\langle u^\#,v^\#\rangle$ を**依存対**であると呼ぶ．依存対の全体を $DP(R)$ で記す．擬順序 $\gtrsim$ が**弱簡約化順序**であるとは，$\gtrsim$ が文脈と代入に閉じ，$\gtrsim$ が整礎かつ代入に閉じていることである．

命題 2.1 [3] 任意の t に対し $t\gtrsim t^\#$ となる弱簡約化順序 $\gtrsim$ が存在して $R\subseteq\gtrsim$ かつ $DP(R)\subseteq\gtrsim$ が成立するならば STRS R は停止性を持つ．

擬簡約性は容易に判定可能であり，擬簡約性と停止性を持つ STRS は十分完全性を持つ．よって，本命題は十分完全性の証明法を与える．

切り落とし関数 π は，各 $f\in\Sigma$ $(\tau(f)=\alpha_1\to\cdots\to\alpha_n\to\beta)$ を $i_1<\cdots<i_m\le n$ となる正整数のリスト $[i_1,\ldots,i_m]$ に割り当てる関数である．$i_j\le k$ を満たす $\pi(f)$ の最大部分リスト $[i_1,\ldots,i_j]$ を $\pi(f)^{\le k}$ で表す．項 $a(t_1,\ldots,t_n)$ に対し $\pi(a(t_1,\ldots,t_n))$ は以下で定義される．

$$\begin{cases} a(\pi(t_1),\ldots,\pi(t_n)) & \text{if } a\in\mathcal{V}\\ a(\pi(t_{i_1}),\ldots,\pi(t_{i_m})) & \text{if } \pi(a)^{\le n}=[i_1,\ldots,i_m]\end{cases}$$

狭義の半順序 $>$ から，$s\gtrsim_\pi t$ を $\pi(s)\ge\pi(t)$ で定義する．なお，TRS においては任意の簡約化順序 $>$(文脈と代入に閉じた整礎な狭義の半順序) に対して $\gtrsim_\pi$ は弱簡約化順序になるが [1]，STRS では，$\gtrsim_\pi$ が弱簡約

化順序になるためには (本論文中の例題は全て満たす) 適当な制限が必要となる [4].

3. 静的再帰構造解析による十分完全性証明法

先ず，再帰構造を再帰対の概念を用いて定式化する.

定義 3.1 (再帰対) $\mathcal{D}_R$ 上の擬順序 $\gtrsim_{fc}$ は次で定義する関係 $\gtrsim'_{fc}$ の反射推移閉包として定義する.

• $f \gtrsim'_{fc} g \overset{\text{def}}{\Longleftrightarrow} f(l_1,\ldots,l_n) \to C[g(r_1,\ldots,r_m)] \in R$

STRS R の再帰対 (recursion pair) 全体からなる集合 $RP(R)$ を以下で定義する.

• $RP(R) \overset{\text{def}}{=} \{\langle u^\#, v^\#\rangle \in DP(R) \mid root(u) \sim_{fc} root(v)\}$

例えば，R_4 の再帰対 $RP(R_4)$ は以下の 2 つになる.

$$\begin{cases} \langle +^\#(s(x),y), +^\#(x,y)\rangle \\ \langle foldl^\#(f,z,cons(x,xs)), foldl^\#(f,f(z,x),xs)\rangle \end{cases}$$

なお，R_4 の依存対 $DP(R_4)$ は $RP(R_4)$ の 2 つに以下の 2 つを加えた 4 つの対になる.

$$\begin{cases} \langle foldl^\#(f,z,cons(x,xs)), f(z,x)\rangle \\ \langle sum^\#(xs), foldl^\#(+,0,xs)\rangle \end{cases}$$

再帰対は静的な再帰構造に対応し高階変数を考慮する必要がないので，依存対 $\langle foldl^\#(f,z,cons(x,xs)), f(z,xs)\rangle$ は $RP(R)$ に含まれない．次に本節で提案する証明法の土台をなす強計算性の概念を与える.

定義 3.2 (強計算性) STRS R において，強計算性を持ち型 α を持つ項の集合 $SCT_\alpha(R)$ を以下のように単純型の構成に基づき帰納的に定義する．なお，$\alpha = \alpha_1 \to \cdots \to \alpha_n \to \beta$ かつ $\beta \in \mathcal{B}$ であるとする.

$$t \in SCT_\alpha(R) \overset{\text{def}}{\Longleftrightarrow} \forall t_i \in SCT_{\alpha_i}(R) \cap Val(R), SC_{cbv}(t(t_1,\ldots,t_n))$$

ここで，$SCT(R)$ を $\bigcup_{\alpha\in\mathcal{S}} SCT_\alpha(R)$ で，$SCV(R)$ を $SCT(R) \cap Val(R)$ で定義する.

定義より直ちに次の補題が導かれる.

補題 3.3

(1) $t \in \mathcal{T}_\tau^{\mathcal{B}}(\Sigma,\mathcal{V})$ かつ $\exists t'(t \overset{*}{\to}_{cbv} t' \in SCT(R))$ ならば $t \in SCT(R)$.

(2) $t \in SCT(R)$ かつ $t_1,\ldots,t_n \in SCV(R)$ ならば $t(t_1,\ldots,t_n) \in SCT(R)$

(3) $t \in SCT(R)$ ならば，ある $c \in SCV(R)$ が存在して $t \overset{*}{\to}_{cbv} c$.

定義 3.4 (基本型限定再帰構造) STRS R が基本型限定再帰構造を持つとは，全ての $\langle u^\#, v^\#\rangle \in RP(R)$ について $\tau(v^\#) \in \mathcal{B}$ が成立することである.

定義 3.5 (高階成長性) STRS R が高階成長性を持つとは，全ての $a(l_1,\ldots,l_n) \to r \in R$ と r 中に出現する高階変数 z に対し，ある i で $l_i \equiv z$ となることである.

定義 3.6 (比較引数) 比較 $s \gtrsim_\pi a(t_1,\ldots,t_m)$ において $i \in \pi(a)$ であるとき t_i を比較引数と呼ぶ.

定義 3.7 (堅固) 項 t が堅固 (firmness) であるとは，t におけるどの変数の出現も引数を持たないことである.

補題 3.8 $u^\# >_\pi v^\#$ が成立し，全ての比較引数が堅固な構成子項であるとする．このとき任意の値代入 $\tilde{\theta}$ に対し，$v^\#\tilde{\theta} \overset{*}{\to}_{cbv} c \Rightarrow \pi(v^\#\tilde{\theta}) \equiv \pi(c)$ が成立.

補題 3.9 $\gtrsim_\pi$ を弱簡約化順序とし，基本型限定再帰構造と高階成長性を持つ STRS R が以下を満たすとする.

(1) 任意の $\langle u^\#, v^\#\rangle \in RP(R)$ に対して，$u^\# \gtrsim_{\sim\pi} v^\#$ が成立し，全ての比較引数が堅固な構成子項.

(2) $QR(R)$

$a \in \mathcal{D}_R, c_i \in SCV(R), \tau(a) = \alpha_1 \to \cdots \to \alpha_m \to \beta, n(\le m)$ とすると $a(c_1,\ldots,c_n) \in SCT(R)$ が成立.

証明 $\tau(u) = \alpha_1 \to \cdots \to \alpha_n \to \alpha$ かつ $\alpha \in \mathcal{B}$ であるとき，$ar(u) = n$ と記すことにする．$t \equiv a(c_1,\ldots,c_n)$ とする．項上の順序 $\blacktriangleright$ を，$>_{fc}$，自然数上での大小比較，$\gtrsim_\pi$ の辞書式結合によって定義し，$\blacktriangleright$ を用いて $(a, ar(t), t^\#)$ に関する帰納法で $t \in SCT(R)$ を示す.

• $\tau(t) = \alpha_1 \to \cdots \to \alpha_m \to \beta$ の場合.

各 i で $c_{n+i} \in SCV(R)$ とし $t' = t(c_{n+1},\ldots,c_{n+m})$ とすると，$root(t) \sim_{fc} root(t')$ かつ $ar(t) = m > 0 = ar(t')$ であるので，$t \blacktriangleright t'$ となる．故に，帰納法の仮定から $t' \in SCT(R)$．よって $t \in SCT(R)$.

• $\tau(t) \in \mathcal{B}$ の場合.

各 c_i は値であるので，$QR(R)$ より，ある $l \to r \in R$ と値代入 $\tilde{\theta}$ が存在して，$t \equiv l\tilde{\theta} \to_{cbv} r\tilde{\theta} \equiv t'$．補題 3.3(1) より，$t' \in SCT(R)$ であることが示せれば十分．$\forall r' \in Sub(r).r'\tilde{\theta} \in SCT(R)$ であることを r' に関する帰納法で示すことにより $t' \equiv r\theta \in SCT(R)$ を示す.

$r' = a'(r'_1,\ldots,r'_m)$ とする．内側の帰納法の仮定より $\forall i.r'_i\tilde{\theta} \in SCT(R)$ であるので，ある $c'_1,\ldots,c'_m \in Val(R)$ が存在して，$r'\tilde{\theta} \equiv (a'\tilde{\theta})(r'_1\tilde{\theta},\ldots,r'_m\tilde{\theta}) \overset{*}{\to}_{cbv} (a'\tilde{\theta})(c'_1,\ldots,c'_m)$．以下では $(a'\tilde{\theta})(c'_1,\ldots,c'_m) \in SCT(R)$ を示す．これにより題意が導かれる.

(i) $a' \in \mathcal{D}_R, \tau(r') \in \mathcal{B}, a \sim_{fc} a'$ の場合.

$a'\tilde{\theta} \equiv a'$ である．$\langle l^\#, r'^\#\rangle \in RP(R)$ であるので，条件 (1) より $l^\# \gtrsim_{\sim\pi} r'^\#$ となり，$l^\#\tilde{\theta} \gtrsim_{\sim\pi} r'^\#\tilde{\theta}$ が成立．また，条件 (1) と補題 3.8 より $\pi(r'^\#\tilde{\theta}) \equiv \pi(a'^\#(c'_1,\ldots,c'_m))$．よって，$l^\#\tilde{\theta} \equiv a^\#(c_1,\ldots,c_n) \gtrsim_{\sim\pi} a'^\#(c'_1,\ldots,c'_m)$ となり $a(c_1,\ldots,c_n) \blacktriangleright a'(c'_1,\ldots,c'_m)$ となる．よって外側の帰納法の仮定より $a'(c'_1,\ldots.c'_m) \in SCT(R)$.

(ii) $a' \in \mathcal{V}_h$ の場合.

R は高階成長性を持つので，$a' \equiv l_i$ となる i が存在する．$a'\tilde{\theta} \equiv c_i \in SCV(R) \subseteq SCT(R)$ であるので補題 3.3(2) より $(a'\tilde{\theta})(c'_1,\ldots.c'_m) \in SCT(R)$ が成立.

(iii) $a' \in \mathcal{D}_R$ かつ (i) 以外の場合.

R は基本型限定再帰構造を持つことを考えて $a >_{fc} a'$ が成立．よって，帰納法の仮定より題意成立.

(iv) $a' \in \mathcal{C}_R$ または $a' \in \mathcal{V} \wedge \tau(a') \in \mathcal{B}$ の場合.

$(a'\tilde{\theta})(c'_1,\ldots,c'_m) \in Val(R)$ なので題意成立. □

定理 3.10 $\gtrsim_\pi$ をある弱簡約化順序とし，基本型限定再帰構造と高階成長性を持つ STRS R が以下を満たすとする．このとき，$SC_{cbv}(R)$ が成立.

(1) 任意の $\langle u^\#, v^\#\rangle \in RP(R)$ に対して，比較 $u^\# \gtrsim_{\sim\pi} v^\#$ が成立し，全ての比較引数が堅固な構成子項.

(2) $QR(R)$

証明 補題3.3(3)より，$\forall t \in \mathcal{T}_r(\Sigma).\, t \in SCT(R)$を示せば良い．これは，補題3.9を用いることにより，tの構造に関する帰納法で容易に証明できる． □

本定理を用いると，文献[4]で提案された再帰経路順序ベースの$\gtrsim_\pi^{rpo}$を用いて，1節で与えたR_1からR_5までの全てのSTRSの十分完全性を示すことができる．例えば，$\pi(+^\#) = [1], \pi(foldl^\#) = [3]$とすることにより，全ての$RP(R_4)$の再帰対に対し以下のような順序付けが成立し，$SC_{cbv}(R_4)$を証明することができる．

$$\pi(+^\#(s(x), y)) \equiv +^\#(s(x)) >^{rpo} +^\#(x) \equiv \pi(+^\#(x, y))$$

$$\pi(foldl^\#(f, z, cons(x, xs))) \equiv foldl^\#(cons(x, xs)) >^{rpo} foldl^\#(xs) \equiv \pi(foldl^\#(f, z, xs))$$

この例は，我々の手法が高階変数の影響を事実上除去できる事を示している．

一方，定理3.10は比較引数として堅固な構成子項しか使えない．R_1からR_5までのSTRSは構成子上の再帰で定義されていたため定理3.10が適用できたが，減算のようななんらかの意味で減少する関数を用いた再帰構造には適用できない．例えば，次のSTRS R_7は減算を用いて除算の定義を与える．なお，$\bot$は0による除算エラーを表現する．

$$\left\{\begin{array}{rcll} -(x,0) & \to & x, & -(\bot, y) \to \bot \\ -(0,y) & \to & 0, & -(x, \bot) \to \bot \\ -(s(x), s(y)) & \to & -(x,y), & \\ \div(x,0) & \to & \bot, & \div(x, \bot) \to \bot \\ \div(0, s(y)) & \to & 0, & \div(\bot, y) \to \bot \\ \div(\underline{s(x)}, s(y)) & \to & s(\div(\underline{-(x,y)}, s(y))) & \end{array}\right\}$$

$RP(R_7)$は以下のようになる．

$$\left\{\begin{array}{l} \langle -^\#(s(x), s(y)), -^\#(x,y)\rangle \\ \langle \div^\#(s(x), s(y)), \div^\#(-(x,y), s(y))\rangle \end{array}\right\}$$

残念ながら定理3.10は$SC_{cbv}(R_7)$を示すことができない．これは，$\div$の定義中の再帰構造において構成子項上の再帰でなく減算$-$を用いて再帰をかけている(上式の下線部)ためである．次に，$SC_{cbv}(R_7)$を示すために非増加関数の概念を提案する．

定義 3.11 (非増加的) $f \in \Sigma$が，$\gtrsim_\pi$に対し非増加関数記号であるとは，$f \in \mathcal{C}_R$であるか，または$f \in \mathcal{D}_R$かつ$root(l) \lesssim_{fc} f$である任意の$l \to r \in R$に対し，rが堅固かつ$l \gtrsim_\pi r$を満たす$\gtrsim_\pi$が存在することである．また，項tが$\gtrsim_\pi$に対し非増加的であるとは，tが堅固でありtに出現する関数記号が全て非増加関数記号であることである．

補題 3.12 項tが$\gtrsim_\pi$に対し非増加的であるとき，どの値代入$\tilde{\theta}$に対しても$t\tilde{\theta} \xrightarrow{*}_{cbv} c \in Val(R) \Longrightarrow t\tilde{\theta} \gtrsim_\pi c$.

証明 $t\tilde{\theta} \to^n_{cbv} c \in Val(R)$として，$n$と$|t|$に関する2重帰納法で証明できる． □

定理 3.13 $\gtrsim_\pi$をある弱簡約化順序とし，基本型限定再帰構造と高階成長性を持つSTRS Rが以下を満たすとする．このとき，$SC_{cbv}(R)$が成立．

(1) 任意の$\langle u^\#, v^\#\rangle \in RP(R)$に対して，比較$u^\# \gtrsim_\pi v^\#$が成立し，その全ての比較引数が非増加的．

(2) $QR(R)$

証明 補題3.8を補題3.12に置き換えることにより定理3.10と同様に証明できる． □

本定理を用いると$SC_{cbv}(R_7)$を示すことができる．実際，文献[4]で提案された再帰経路順序ベースの$\gtrsim_\pi^{rpo}$を用いると，$\pi(-^\#) = \pi(-) = \pi(\div^\#) = [1]$とし，関数記号の重み付けを$s \rhd -$とすることにより以下のように$RP(R_7)$に対して順序付けが成功する．

$$\pi(-^\#(s(x), s(y))) \equiv -^\#(s(x)) >^{rpo} -^\#(x) \equiv \pi(-^\#(x, y))$$

$$\pi(\div^\#(s(x), s(y))) \equiv \div^\#(s(x)) >^{rpo} \div^\#(-(x)) \equiv \pi(\div^\#(-(x,y), s(y)))$$

4. 終りに

本論文では，高階変数の考慮を必要としない静的再帰構造の解析による十分完全性証明法を与えた．また，この証明法が有効に機能するための四条件(1節中の(II–V))を与えた．本論文で提案した手法は再帰解析において高階変数を考慮必要がないため，次のようなSTRSの十分完全性証明に特に有効である．

$$R_8 = \{twice(f,x) \to f(f(x))\}$$
$$R_9 = \{S(f,g,x) \to f(x, g(x)),\ K(x,y) \to x\}$$

実際，$RP(R_8) = RP(R_9) = \emptyset$なので，定理3.10から直ちに$SC_{cbv}(R_8)$と$SC_{cbv}(R_9)$が示せる．さらに，静的構造のみに着目した手法であるため，$SC_{cbv}(R)$が定理3.10により示された場合，$SC_{cbv}(R \cup R_8)$と$SC_{cbv}(R \cup R_9)$も示せることになる．これは，R_8, R_9の右辺に被定義記号が出現しないためRPと$>_{fc}$を変化させないからである．このように我々が提案した手法はモジュラー性や階層性(hierarchical combination)と呼ばれる分割統治法に相性が良いと考えられる．一方，Rの追加により非増加関数は変化する可能性があるため定理3.13の適用にはより詳細な解析が必要となる．また，本論文で提案した十分完全性証明法を最内停止性証明法や停止性証明法に一般化することも可能だと思われる．これらの研究も今後の課題である．

謝辞: 本研究は一部，科研費#15500007,#16300005，人工知能研究振興財団，名古屋大学21世紀COEプログラム(社会情報基盤のための音声・映像の知的統合)の補助を受けている．

参考文献

[1] T.Arts, J.Giesl, Termination of Term Rewriting Using Dependency Pairs, Theoretical Computer Science, Vol.236, pp.133–178, 2000.

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

[3] Kusakari,K., On Proving Termination of Term Rewriting Systems with Higher-Order Variables, IPSJ Transactions on Programming, Vol.42, No.SIG 7 (PRO 11), pp.35–45, 2001.

[4] Kusakari,K., Higher-Order Path Orders based on Computability, IEICE Transactions on Information and Systems, Vol.E87-D, No.2, pp.352–359, 2004.

[5] W.W.Tait, Intensional Interpretation of Functionals of Finite Type, Journal of Symbolic Logic 32, pp.198–212, 1967.