情報処理学会論文誌　プログラミング　Vol. 4　No. 2　13–30 (Mar. 2011)

例外処理を持つ関数型プログラムの停止性・非停止性証明法

濱口　毅[†1]　酒井正彦[†1]
馬場正貴[†1,*1]　阿草清滋[†1]

本論文では，例外処理を持つ先行評価に基づく関数型プログラムの停止性・非停止性証明法を提案する．停止性と非停止性を保存する関数型プログラムの文脈依存項書き換え系（CS-TRS）への変換法を与える．これにより，近年開発が進んでいる既存のCS-TRS の停止性証明ツールを用いることが可能になる．先行評価での実行においては，関数の引数を先頭から順に評価してから関数の値を計算するため，生成される書き換え系においては，この評価は基本的には最内評価に対応する．しかし，停止性を持たない引数と例外を発生させる引数の順序によって停止性に影響を与えるため，引数の評価順を厳密に考慮する必要がある．そのため，文脈依存の機能を用いて引数の評価順を制御する．また，例外が発生した場合には，それ以降の評価を止めて，例外が処理されるまで例外値を戻す必要がある．これを行えるように書き換え規則を生成する．また，この変換の健全性，すなわち変換後の CS-TRS が最内停止性を持てばプログラムも停止性を持つことと，完全性，すなわちプログラムが停止性を持つならば変換後の CS-TRS も最内停止性を持つことを示す．これにより，変換後の CS-TRS の最内（非）停止性が示すことができれば，プログラムの（非）停止性が証明されることが保証される．

Proving Method of Termination/Non-Termination for Functional Programs with Exception Handling

Takeshi Hamaguchi,[†1] Masahiko Sakai,[†1]
Masataka Baba[†1,*1] and Kiyoshi Agusa[†1]

We present a proving method of termination/non-termination for functional programs based on eager-evaluation with exception handling. We give a transformation of functional programs into Context-Sensitive Term Rewriting Systems (CS-TRSs). This enables us to use recent-developed termination provers for CS-TRSs as termination provers for functional programs. Since eager-evaluation computes a return value of a function after all values of its arguments are computed in right-to-left order, this evaluation corresponds to the innermost reduction on generated TRSs. The termination property is affected by the occurrences of non-terminating argument and exception-raising argument. This means that we have to handle the evaluation order strictly. Thus we used context sensitiveness for this purpose and design the transformation producing rewrite rules, that repeatedly pass back an exception to the caller until it is handled when exception occurs. We prove the soundness and completeness of the transformation, that is, the innermost termination of the CS-TRS implies the termination of the program and vise versa. This allows us to prove (non-)termination of programs by proving (non-)termination of CS-TRSs.

1. はじめに

多くのプログラミング言語には例外処理機構が用意されており，例外発生時に適切な処理を行ったうえでプログラムを終了させたり例外から回復したりすることができる．例外処理は一種のジャンプであり，通常の処理と制御の流れが異なり，その動作の解析は困難である．これまでに例外処理を持つプログラムを定式化するさまざまなモデルが提案されている[1)–3)]．これらのモデルは主に例外処理を持つプログラムの変換を目的としている．

例外処理記述が不適切である場合，プログラムが意図したとおりの動作をせず，停止しない場合がある．例外を用いたプログラムでは，意図したとおりに例外をとらえてプログラムが停止するかどうかの検証が重要であるが，既存のモデルではプログラムの停止性を判定することができない．例外処理を含む簡単な Standard ML のプログラムとして**図 1** を考える．このプログラムの関数 `f` が適用されると `f(0)` の評価が際限なく行われるように見えるが，実際は引数の値にかかわらず評価は停止する．なぜならばこの関数が適用されるとまず `raise A` が評価され，`f(0)` を評価する前に例外ハンドラによって例外処理式が評価されるからである．このようなプログラムの停止性を正しく判定することが求められる．

これまでに我々は，ML/ex プログラムという例外処理機能を持つ先行評価に基づく単純な関数型プログラムを文脈依存項書き換え系（CS-TRS）に変換する方法を与えている[4)]．これにより得られた CS-TRS は，それが最内停止性を持つなら変換前のプログラムも停止

†1 名古屋大学
Nagoya University
*1 現在，KDDI 株式会社
Presently with KDDI Corporation

```
exception A
exception B
fun f x =  ((raise A) + f(0)) handle A => 1
```

図 1　例外処理を含む停止する SML プログラム
Fig. 1　SML program with termination including exception handling.

性を持つように変換されている，すなわち健全性を持つ．そのため，近年さかんに開発が進んでいる AProVE[5] や μ-Term[6] 等の停止性証明ツールを用いて，得られた CS-TRS の停止性を示すことでプログラムの停止性を示すことができる．しかしこの手法では完全性，すなわちプログラムが停止性を持つならば変換後の CS-TRS も最内停止性を持つという性質が成り立たたず，変換後の CS-TRS が最内停止性を持たないことが分かっても，プログラムが停止性を持たないとは限らない．

本論文ではこの変換を，健全性だけでなく完全性を持つように改良する．また，この変換による CS-TRS の計算結果とプログラムの実行結果が等しいことを示し，それを利用して健全性と完全性を持つことを証明する．

2. 準　　備

本章では文脈依存項書き換え系（CS-TRS）を与える．

定義 2.1（項）　N を自然数の集合とする．可算無限個の変数の集合 $\mathcal{V}$，有限個の関数記号の集合 $\mathcal{F}$，写像 $arity : \mathcal{F} \to N$ から生成されるすべての項の集合 $\mathcal{T}(\mathcal{F},\mathcal{V})$ は，以下のように再帰的に定義される．

- $\mathcal{V} \subseteq \mathcal{T}(\mathcal{F},\mathcal{V})$
- $f \in \mathcal{F}$, $arity(f) = n\ (\geq 0)$ かつ $t_1,\ldots,t_n \in \mathcal{T}(\mathcal{F},\mathcal{V})$ ならば，$f(t_1,\ldots,t_n) \in \mathcal{T}(\mathcal{F},\mathcal{V})$

特に変数を含まない項の集合を $\mathcal{T}(\mathcal{F})$，項 $t_1,\ldots,t_n$ のリストを $[t_1,\ldots,t_n]$ と表す．項 t に現れるすべての変数の集合を $\mathcal{V}ar(t)$ で表す．

定義 2.2（代入）　変数から項への写像 $\theta : \mathcal{V} \to \mathcal{T}(\mathcal{F},\mathcal{V})$ はその定義域 $\mathcal{D}om(\theta) = \{x \mid x \in \mathcal{V}, x \neq \theta(x)\}$ が有限集合であるとき代入であるという．代入 θ は項から項への写像に自然に拡張される．すなわち，$\theta(t)$ は項 t 中の各変数 x を項 $\theta(x)$ に置き換えて得られる項である．特に，どの $x \in \mathcal{D}om(\theta)$ に対しても $\theta(x) \in \mathcal{T}(\mathcal{F})$ であるとき，θ を基底代入という．また，定義域が空の代入を $\emptyset$ と書く．なお，$\theta(t)$ は $t\theta$ と略記する．代入 θ は項のリストから項のリストへの写像に自然に拡張される．すなわち，$[t_1,\ldots,t_n]\theta$ はリスト中の各項 t_i を $t_i\theta$ に置き換えて得られるリストである．また，$[\,]\theta = [\,]$ とする．

定義 2.3（項の位置集合）　項 t の位置の集合 $\mathcal{P}os(t)$ は次のように再帰的に定義される．

- $t \in \mathcal{V}$ ならば $\mathcal{P}os(t) = \{\varepsilon\}$
- $t = f(t_1,\ldots,t_n)$ ならば $\mathcal{P}os(t) = \{\varepsilon\} \cup \bigcup_{i=1}^{n}\{ip \mid p \in \mathcal{P}os(t_i)\}$

位置 p にある t の部分項を $t|_p$ で表す．項 t の任意の p 部分項を t' で置き換えて得られる項を $t[t']_p$ と表す．また，$t = t[t']_p$ のとき，t' は t の部分項であり，$t \trianglerighteq t'$ または $t' \trianglelefteq t$ と書く．特に $p \neq \varepsilon$ のとき，t' は t の真部分項であり，$t \triangleright t'$ または $t' \triangleleft t$ と書く．$root(t)$ は項 t の先頭，すなわち位置 ε の関数記号を表す．

定義 2.4（文脈依存項書き換え系）　$\mathcal{F}$ 上の書き換え規則 (l,r) は $l \notin \mathcal{V}$，$l,r \in \mathcal{T}(\mathcal{F},\mathcal{V})$，$\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$ を満たす項の対 l，r であり，$l \to r$ で表す．$\mathcal{F}$ 上の文脈依存項書き換え系（CS-TRS）は $\mathcal{F}$ 上の書き換え規則の集合 $\mathcal{R}$ と文脈 μ の組 $\langle\mathcal{R},\mu\rangle$ である．ここで μ は，どの $arity(f) > 0$ となる $f \in \mathcal{F}$ に対しても $\mu(f) \subseteq \{1,\ldots,arity(f)\}$ を満たす写像である．項 t に対する μ 置換位置 $\mathcal{P}os^{\mu}(t)$ は次のように再帰的に定義される．

- $t \in \mathcal{V}$ ならば $\mathcal{P}os^{\mu}(t) = \{\varepsilon\}$
- $t = f(t_1,\ldots,t_n)$ ならば $\mathcal{P}os^{\mu}(t) = \{\varepsilon\} \cup \{ip \mid i \in \mu(f), p \in \mathcal{P}os^{\mu}(t_i)\}$

また，項 t に現れる μ 置換のすべての変数の集合は $\mathcal{V}ar^{\mu}(t) = \{x \mid \exists p \in \mathcal{P}os^{\mu}(t), t|_p = x \in \mathcal{V}ar(t)\}$ と定義される．ある $l \to r \in \mathcal{R}$ について，$t = t[l\theta]_p$ かつ $p \in \mathcal{P}os^{\mu}(t)$ を満たすとき，$l\theta$ は文脈依存リデックスという．このとき，$l\theta$ が他の文脈依存リデックスを持たないとき，$l\theta$ は t の文脈依存最内リデックスという．文脈依存項書き換え系 $\langle\mathcal{R},\mu\rangle$ に対して，項 s の文脈依存最内リデックスが $l\theta$ であるとき，$s = s[l\theta]_p$ は $t = s[r\theta]_p$ に最内書き換えされるといい，$s \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} t$ と書く．また，文脈依存リデックスを持たない項を文脈依存正規形（以下では単に正規形）という．また，$s \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} t$ かつ t が正規形であるとき，t は s の最内正規形という．$\underset{in}{\to}^{+}_{\langle\mathcal{R},\mu\rangle}$ は $\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ の推移閉包，$\underset{in}{\to}^{*}_{\langle\mathcal{R},\mu\rangle}$ は $\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ の反射推移閉包を表す．$s = s[t]_p$ かつ $p \in \mathcal{P}os^{\mu}(s)$ のとき，t は s の文脈依存部分項であり $s \trianglerighteq_{\mu} t$ と書く．CS-TRS $\langle\mathcal{R},\mu\rangle$ に対して，$t \in \mathcal{T}(\mathcal{F},\mathcal{V})$ から始まる無限の $\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ 書き換え系列が存在しないとき，t は $\langle\mathcal{R},\mu\rangle$ 最内停止性を持つといい，どの項も $\langle\mathcal{R},\mu\rangle$ 最内停止性を持つとき，$\langle\mathcal{R},\mu\rangle$ は最内停止性を持つという．任意の異なる規則 $l_1 \to r_1$，$l_2 \to r_2 \in \mathcal{R}$ について，$l_1\theta_1 = l_2\theta_2$ となる代入 θ_1，θ_2 が存在しないとき，CS-TRS $\langle\mathcal{R},\mu\rangle$ は根重なりがないという．

3. 関数型プログラミング言語 ML/ex

本章では変換対象となる関数型プログラミング言語 ML/ex の定義と意味論を与える．ML/ex は以下のように定義される[4]．

定義 3.1（ML/ex） ML/ex のプログラムを $\mathcal{P} = \langle \mathcal{F}, \mathit{Fundef}, \mathit{Except} \rangle$ と定義する．$\mathcal{F}$ は関数記号の集合であり，$\mathcal{F} = \mathcal{F}_\mathcal{C} \cup \mathcal{F}_\mathcal{B} \cup \mathcal{F}_\mathcal{D} \cup \mathit{Except}$，$\mathcal{F}_\mathcal{C} = \{0, \mathit{succ}, \mathit{True}, \mathit{False}\}$，$\mathcal{F}_\mathcal{B} = \{+, *, \mathit{div}, \mathit{mod}, \mathit{not}, <=, <, >=, >, =, <>, \mathit{if}, \mathit{raise}, \mathit{handle}\}$ である．$\mathcal{F}_\mathcal{D}$ はプログラムで定義される関数記号の集合である．*Fundef* は $\mathcal{F}$ 上の書き換え規則の集合，*Except* は例外の集合である．

$\mathcal{F}_\mathcal{C}$ は真理値型と自然数型の値を表現するための関数記号の集合，$\mathcal{F}_\mathcal{B}$ は真理値型と自然数型の基本的な演算の組み込み関数記号の集合である．また，$\mathcal{F}_{\mathcal{B}'} = \mathcal{F}_\mathcal{B} \backslash \{\mathit{if}, \mathit{handle}, \mathit{raise}\}$ とし，$\mathcal{F}_{\mathcal{B}'}$ の関数を定義する書き換え規則の集合を *BuiltinDef* とする．たとえば関数 + に関する書き換え規則は以下のとおりである．

$$\{+(0, y) \to y, +(\mathit{succ}(x), y) \to \mathit{succ}(+(x, y))\} \subseteq \mathit{BuiltinDef}$$

他の組み込み関数についても根重なりがない規則集合で表されるが，本論文では記述を割愛する．

図 1 の SML プログラムを ML/ex で記述すると**図 2** のようになる．ただし，関数記号の集合 $\mathcal{F}$ において *Fundef* で使用しない組み込み関数記号は省略した．

ML/ex の意味論は ML/ex の式の値を求める関数 *ev* により与えられる．関数 *ev* は 1 つの項 t と 1 つの代入 θ を引数としてとり，1 つの値 M を返す．関数 *evArgs* は 1 つの項のリスト t と 1 つの代入 θ を引数としてとり，値 M のリストを返す．任意の項 t について，$\mathit{ev}(t, \theta)$ によって値が求まるならその値が一意に定まるように関数 *ev* と *evArgs* の定義を与える．値 M は例外値または正常値またはその他の値である．$\downarrow$ の付いた値はこれ以上評価のできない正常値であることを表し，$\uparrow$ は値が例外値であることを表す．また，$\bot$ はその他の値であることを表す．$\mathit{ev}(t, \theta)$ の値が定まらないとき，その値は未定義であるという．直感的には t の計算が停止しないプログラムに対応する．

$$\mathcal{P} = \langle \{0, \mathit{succ}, +, \mathit{raise}, \mathit{handle}, f\}, \mathit{Fundef}, \{A, B\} \rangle$$
$$\mathit{Fundef} = \{f(x) \to \mathit{handle}(+(\mathit{raise}(A), f(0)), A, \mathit{succ}(0))\}$$

図 2　例外処理を含む停止する ML/ex プログラム（1）
Fig. 2　ML/ex program with termination including exception handling (1).

なお，以下では代入 θ は $\mathcal{V}$ から $\mathcal{T}(\mathcal{F}_\mathcal{C})$ への写像であるとする．

定義 3.2（関数 *match*） 関数 *match* は次のように定義される．各 i（$1 \le i \le n$）について，$t_i \in \mathcal{T}(\mathcal{F})$，$l_i \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ において $t_i = l_i\sigma$ となる代入 σ が存在するとき，$\mathit{match}[t_1, \ldots, t_n][l_1, \ldots, l_n] = \sigma$ とする．

定義 3.3（関数 *ev*） 関数 *ev* は以下のように定義される．

- $t = x \in \mathcal{V}$ のとき $\mathit{ev}(t, \theta) = x\theta\downarrow$
- $t = \mathit{succ}(t')$ のとき，
  - $\mathit{ev}(t', \theta) = v\downarrow$ ならば $\mathit{ev}(t, \theta) = \mathit{succ}(v)\downarrow$
  - $\mathit{ev}(t', \theta) = E\uparrow$ ならば $\mathit{ev}(t, \theta) = E\uparrow$
  - $\mathit{ev}(t', \theta) = \bot$ ならば $\mathit{ev}(t, \theta) = \bot$
- $t = c \in (\mathcal{F}_\mathcal{C} \backslash \{\mathit{succ}\})$ のとき $\mathit{ev}(t, \theta) = c\downarrow$
- $t = \mathit{raise}(t')$ のとき，
  - $t' = E \in \mathit{Except}$ のとき，$\mathit{ev}(t, \theta) = E\uparrow$
  - $t' \notin \mathit{Except}$ のとき，$\mathit{ev}(t, \theta) = \bot$
- $t = \mathit{handle}(t_1, t_2, t_3)$ のとき，
  - $t_2 = E \in \mathit{Except}$ のとき，
    - $\mathit{ev}(t_1, \theta) = v\downarrow$ ならば $\mathit{ev}(t, \theta) = v\downarrow$
    - $\mathit{ev}(t_1, \theta) = E\uparrow$ ならば $\mathit{ev}(t, \theta) = \mathit{ev}(t_3, \theta)$
    - $E' \in \mathit{Except}$ かつ $E \neq E'$ かつ $\mathit{ev}(t_1, \theta) = E'\uparrow$ ならば $\mathit{ev}(t, \theta) = E'\uparrow$
    - $\mathit{ev}(t_1, \theta) = \bot$ ならば $\mathit{ev}(t, \theta) = \bot$
  - $t_2 \notin \mathit{Except}$ のとき，$\mathit{ev}(t, \theta) = \bot$
- $t = \mathit{if}(t_1, t_2, t_3)$ のとき，
  - $\mathit{ev}(t_1, \theta) = \mathit{True}\downarrow$ ならば $\mathit{ev}(t, \theta) = \mathit{ev}(t_2, \theta)$
  - $\mathit{ev}(t_1, \theta) = \mathit{False}\downarrow$ ならば $\mathit{ev}(t, \theta) = \mathit{ev}(t_3, \theta)$
  - $\mathit{ev}(t_1, \theta) = v\downarrow$ かつ $v \neq \mathit{True}$ かつ $v \neq \mathit{False}$ ならば $\mathit{ev}(t, \theta) = \bot$
  - $\mathit{ev}(t_1, \theta) = E\uparrow$ ならば $\mathit{ev}(t, \theta) = E\uparrow$
  - $\mathit{ev}(t_1, \theta) = \bot$ ならば $\mathit{ev}(t, \theta) = \bot$
- $t = f(t_1, \ldots, t_n)$ かつ $f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_\mathcal{D}$ のとき，
  - $\mathit{evArgs}([t_1, \ldots, t_n], \theta) = [v_1, \ldots, v_n]\downarrow$ かつ $f(l_1, \ldots, l_n) \to r \in \mathit{Fundef} \cup \mathit{BuiltinDef}$ かつ $\sigma = \mathit{match}[v_1, \ldots, v_n][l_1, \ldots, l_n]$ ならば $\mathit{ev}(t, \theta) = \mathit{ev}(r, \sigma)$
  - $\mathit{evArgs}([t_1, \ldots, t_n], \theta) = E\uparrow$ ならば $\mathit{ev}(t, \theta) = E\uparrow$

- $evArgs([t_1,\ldots,t_n],\theta)=\bot$ ならば $ev(t,\theta)=\bot$

- $t\in Except$ のとき $ev(t,\theta)=\bot$

定義 3.4（関数 $evArgs$） 関数 $evArgs$ は以下のように定義される．

- $t=[\,]$ のとき $evArgs([\,],\theta)=[\,]$
- $t=t_1::ts$ のとき，
  - $ev(t_1,\theta)=v\downarrow$ ならば
    - $evArgs(ts,\theta)=vl\downarrow$ ならば $evArgs(t,\theta)=(v::vl)\downarrow$
    - $evArgs(ts,\theta)=E\uparrow$ ならば $evArgs(t,\theta)=E\uparrow$
    - $evArgs(ts,\theta)=\bot$ ならば $evArgs(t,\theta)=\bot$
  - $ev(t_1,\theta)=E\uparrow$ ならば $evArgs(t,\theta)=E\uparrow$
  - $ev(t_1,\theta)=\bot$ ならば $evArgs(t,\theta)=\bot$

定義 3.5（ML/ex の停止性） プログラム $\mathcal{P}$ についてどの基底項 $t\in\mathcal{T}(\mathcal{F})$ についても $ev(t,\emptyset)$ の値が定まるとき，$\mathcal{P}$ は停止性を持つという．

図 2 のプログラムにおいて $ev(f(0),\emptyset)$ の値を求めることができ，その値は $succ(0)\downarrow$ となる．

4. ML/ex の文脈依存項書き換え系への変換

ML/ex から CS-TRS への変換 ϕ を与える．得られる CS-TRS が ML/ex のプログラムの実行を模倣するように ϕ を設計する．先行評価型言語のプログラムの実行においては関数の引数を先頭から順にすべて計算してから関数の値を計算するため，基本的には変換後の書き換え系において項の評価を最内評価で行えばよい．しかしながら，プログラム中の関数の引数に停止しない式と例外を発生する式がともに存在するとき，どちらを先に実行されるかによって結果が異なる．このため引数が先頭から順に評価されるように CS-TRS の文脈による書き換えの制御を行う．すなわち，関数の引数は左から順番に評価し，引数に例外が発生しなかった場合には次の引数を評価する．引数に例外が発生した場合はそれ以降の引数の評価をせずに関数の値を例外値とするような CS-TRS を生成する．

以上の制御を行うため，関数記号 $f\in(\mathcal{F}_{\mathcal{B}}\cup\mathcal{F}_{\mathcal{D}})\backslash\{if,handle,raise\}$ について，$arity(f)=n$ のとき，$\phi(\mathcal{P})$ には関数記号 $f_1,\ldots,f_{n+1}$ が加えられる．f_i（$1\le i\le n$）はそれぞれ f の第 i 引数を評価し，その結果が例外を引き起こすならただちに関数 f の値を例外値にするための関数である．また，f_{n+1} はすべての引数が正常値である場合に値を求めるための関数である．$\phi(\mathcal{P})$ では新たな定数記号 tt，関数記号 $fire$，$guard$，$select$，$isData$ を導入する．

定義 4.1（ML/ex から CS-TRS への変換） $\mathcal{P}=\langle\mathcal{F},Fundef,Except\rangle$ を ML/ex プログラムとする．ここで $\mathcal{F}=\mathcal{F}_{\mathcal{B}}\cup\mathcal{F}_{\mathcal{C}}\cup\mathcal{F}_{\mathcal{D}}\cup Except$ である．$\mathcal{F}_{\mathcal{B}'}=\mathcal{F}_{\mathcal{B}}\backslash\{if,handle,raise\}$ とするとき，以下のように $\mathcal{F}_{\mathcal{E}}$ を定義する．

$$\mathcal{F}_{\mathcal{E}}=\{f_1,\ldots,f_{n+1}\mid f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}},arity(f)=n\}$$

このとき，$\mathcal{F}'=\mathcal{F}\cup\mathcal{F}_{\mathcal{E}}\cup\{tt,fire,guard,select,isData\}$ とする．変換 ϕ は $\mathcal{P}$ を入力とし，$\mathcal{F}'$ 上の CS-TRS $\langle\mathcal{R},\mu\rangle$ を出力とする．すなわち $\phi(\mathcal{P})=\langle\mathcal{R},\mu\rangle$ である．$\mathcal{F}'$ に対する μ は，$\mu(raise)=\mu(fire)=\mu(isData)=\emptyset$，$\mu(succ)=\mu(if)=\mu(handle)=\mu(guard)=\mu(select)=\{1\}$ である．$f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}}$ については $\mu(f)=\emptyset$ である．$f_i\in\mathcal{F}_{\mathcal{E}}$（$1\le i\le n+1$）に対する μ は $1\le i\le n$ のとき $\mu(f_i)=\{i\}$ であり，$\mu(f_{n+1})=\emptyset$ である．$f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}}$ に対して規則集合 $\mathcal{R}_{f0}$ を以下のように定義する．

$$\mathcal{R}_{f0}=\{f(x_1,\ldots,x_n)\to f_1(x_1,\ldots,x_n)\mid f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}}\}$$

$f_i\in\mathcal{F}_{\mathcal{E}}$ に対して規則集合 $\mathcal{R}_f$ を以下のように定義する．

$$\mathcal{R}_f=\{f_i(x_1,\ldots,x_n)\to guard(isData(x_i),f_{i+1}(x_1,\ldots,x_n))|f_i\in\mathcal{F}_{\mathcal{E}},1\le i\le n\}$$

$f=root(l),f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}}$ である $l\to r\in Fundef\cup BuiltinDef$ について，l に現れる f を f_{n+1} に置き換えて得られる項を l' とするとき，その書き換え規則 $l'\to r$ からなる集合を $Fundef'$ とする．規則集合 $\mathcal{R}_E$，$\mathcal{R}_{FE}$，$\mathcal{R}_{CE}$，$\mathcal{R}_{UE}$ を以下のように定義する．

$\mathcal{R}_E=$

$\{guard(tt,y)\to y,\ guard(fire(x),y)\to fire(x),$
$isData(succ(x))\to isData(x),\ isData(0)\to tt,\ isData(True)\to tt,$
$isData(False)\to tt,\ isData(fire(x))\to fire(x),$
$select(tt,x,y,z)\to x,$
$succ(fire(x))\to fire(x),$
$if(True,y,z)\to y,\ if(False,y,z)\to z,\ if(fire(x),y,z)\to fire(x)\}$

$\mathcal{R}_{FE}=\{raise(E)\to fire(E),handle(x,E,z)\to select(isData(x),x,E,z)\mid E\in Except\}$

$\mathcal{R}_{CE}=\{select(fire(E_1),x,E_2,z)\to z\mid E_1\in Except\wedge E_2\in Except\wedge E_1=E_2\}$

$\mathcal{R}_{UE}=\{select(fire(E_1),x,E_2,z)\to fire(E_1)\mid E_1\in Except\wedge E_2\in Except\wedge E_1\neq E_2\}$

$\mathcal{R}=\mathcal{R}_E\cup\mathcal{R}_{FE}\cup\mathcal{R}_{CE}\cup\mathcal{R}_{UE}\cup\mathcal{R}_{f0}\cup\mathcal{R}_f\cup Fundef'$ とする．

図 2 のプログラムを定義 4.1 に従って変換すると**図 3** の CS-TRS になる．ただし $Fundef$ で使用しない関数の $BuiltinDef$ の規則は省略した．図 3 の CS-TRS による $f(0)$ の書き換えのようすを**図 4** に示す．

この変換で生成される CS-TRS $\phi(\mathcal{P})=\langle\mathcal{R},\mu\rangle$ に現れるどの関数 f も $|\mu(f)|\le 1$ を満た

$\mu(raise) = \mu(fire) = \mu(isData) = \mu(+) = \mu(+_3) = \mu(f) = \mu(f_2) = \emptyset$,
$\mu(succ) = \mu(if) = \mu(handle) = \mu(guard) = \mu(select) = \mu(+_1) = \mu(f_1) = \{1\}$,
$\mu(+_2) = \{2\}$
$guard(tt, y) \to y$
$guard(fire(x), y) \to fire(x)$
$isData(succ(x)) \to isData(x)$
$isData(0) \to tt$
$isData(True) \to tt$
$isData(False) \to tt$
$isData(fire(x)) \to fire(x)$
$succ(fire(x)) \to fire(x)$
$if(True, y, z) \to y$
$if(False, y, z) \to z$
$if(fire(x), y, z) \to fire(x)$
$select(tt, x, y, z) \to x$
$raise(A) \to fire(A)$
$raise(B) \to fire(B)$
$handle(x, A, z) \to select(isData(x), x, A, z)$
$handle(x, B, z) \to select(isData(x), x, B, z)$
$select(fire(A), x, A, z) \to z$
$select(fire(B), x, B, z) \to z$
$select(fire(A), x, B, z) \to fire(A)$
$select(fire(B), x, A, z) \to fire(B)$
$+(x, y) \to +_1(x, y)$
$+_1(x, y) \to guard(isData(x), +_2(x, y))$
$+_2(x, y) \to guard(isData(y), +_3(x, y))$
$+_3(0, y) \to y$
$+_3(succ(x), y) \to succ(+(x, y))$
$f(x) \to f_1(x)$
$f_1(x) \to guard(isData(x), f_2(x))$
$f_2(x) \to handle(+(raise(A), f(0)), A, succ(0))$

図 3　ML/ex プログラムを変換した CS-TRS
Fig. 3　CS-TRS transformed from ML/ex program.

す．また，$\mathcal{P}$ の *Fundef* の規則に根重なりがないなら，$\phi(\mathcal{P})$ の規則は根重なりがない．一般にプログラムは根重なりがないプログラムへ変換可能であるため，本論文では根重なりがないプログラムのみを対象とする．根重なりがないプログラムに対する変換の性質として以下の定理が成り立つ．

$f(0) \to f_1(0)$
$\to guard(isData(0), f_2(0))$
$\to guard(tt, f_2(0))$
$\to f_2(0)$
$\to handle(+(raise(A), f(0)), A, succ(0))$
$\to handle(+_1(raise(A), f(0)), A, succ(0))$
$\to handle(+_1(fire(A), f(0)), A, succ(0))$
$\to handle(guard(isData(fire(A)), +_2(fire(A), f(0))), A, succ(0))$
$\to handle(guard(fire(A), +_2(raise(A), f(0))), A, succ(0))$
$\to handle(fire(A), A, succ(0))$
$\to select(isData(fire(A)), fire(A), A, succ(0))$
$\to select(fire(A), fire(A), A, succ(0))$
$\to succ(0)$

図 4　CS-TRS による項の書き換え
Fig. 4　Term Rewriting by CS-TRS.

定理 4.2（プログラムの評価と書き換えの一致）　項 $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$，ならびに $\mathcal{V}ar(t) \subseteq \mathcal{D}om(\theta), \theta : \mathcal{V} \to \mathcal{T}(\mathcal{F}_\mathcal{C})$ を満たす代入 θ に対して以下の（1）と（2）が成り立つ．
（1）　$ev(t, \theta) = v\downarrow$ ならば，かつそのときに限り $t\theta \underset{in}{\to}^*_{\phi(\mathcal{P})} v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$
（2）　$ev(t, \theta) = E\uparrow$ ならば，かつそのときに限り $t\theta \underset{in}{\to}^*_{\phi(\mathcal{P})} fire(E)$
証明は付録 A.1 を参照されたい．

5. ML/ex の停止性・非停止性判定

本章では CS-TRS の停止性判定ツールを利用した，プログラムの停止性判定について述べる．定義 4.1 による CS-TRS への変換においては $\mathcal{P}$ の *Fundef* の規則に根重なりがないなら以下の定理が成り立つ．

定理 5.1（健全性）　CS-TRS $\phi(\mathcal{P})$ が最内停止性を持つならばプログラム $\mathcal{P}$ は停止性を持つ．

定理 5.2（完全性）　プログラム $\mathcal{P}$ が停止性を持つならば CS-TRS $\phi(\mathcal{P})$ は最内停止性を持つ．

なお，定理の証明はそれぞれ付録 A.2，A.3 を参照されたい．この 2 つの定理より変換後の CS-TRS が最内停止性を持つならプログラムも停止性を持ち，CS-TRS が最内停止性を持たないならプログラムも停止性を持たないことがいえる．

本研究では代表的な停止性証明ツールである AProVE [5] と μ-Term [6] を使用し，例外処

$$\mathcal{P} = \langle\{0, succ, +, True, False, if, raise, handle, f\}, Fundef, \{A, B\}\rangle$$
$$Fundef = \{f(x) \to handle(if(True, raise(A), f(x)), A, succ(0))\}$$

図 5　例外処理を含む停止する ML/ex プログラム（2）
Fig. 5　ML/ex program with termination including exception handling (2).

$$\mathcal{P} = \langle\{0, succ, +, True, False, if, raise, handle, f\}, Fundef, \{A, B\}\rangle$$
$$Fundef = \{f(x) \to handle(if(False, raise(A), f(x)), A, succ(0))\}$$

図 6　例外処理を含む停止しない ML/ex プログラム（3）
Fig. 6　ML/ex program with non-termination including exception handling (3).

$$\mathcal{P} = \langle\{0, succ, +, True, False, if, <=, raise, handle, f\}, Fundef, \{A, B\}\rangle$$
$$Fundef = \{f(x) \to handle(if(<= (0, succ(0)), raise(A), f(x)), A, succ(0))\}$$

図 7　例外処理を含む停止する ML/ex プログラム（4）
Fig. 7　ML/ex program with termination including exception handling (4).

$$\mathcal{P} = \langle\{0, succ, +, *, True, False, if, <=, raise, handle, f\}, Fundef, \{A, B\}\rangle$$
$$Fundef = \{f(0) \to succ(0), f(succ(x)) \to *(succ(x), f(x)),\}$$

図 8　例外処理を含まない停止する ML/ex プログラム（5）
Fig. 8　ML/ex program with termination not including exception handling (5).

理を含む ML/ex プログラムの停止性・非停止性の証明を試みた．証明を試みたプログラムは図 2，**図 5**，**図 6**，**図 7**，**図 8** である．図 6 のみ停止性を持たず，その他は停止性を持つプログラムである．また，図 8 のみ例外処理を含まず，その他は例外処理を含むプログラムである．図 2 のプログラムを変換した図 3 の CS-TRS に対してツールを適用した．他のプログラムについても本手法により変換した CS-TRS に対してツールを適用した．ただし図 3 と同様，関数 f の計算で利用されない組み込み関数の規則は省いた．使用したツールは，AProVE Web Interface 版[⋆1]，μ-Term Web Interface 版[⋆2]，ダウンロード版 AProVE1.2 である．AProVE1.2 を実行した計算機の CPU は Intel Core2Duo E6600（2.4 GHz）で，ヒープサイズは 3 G バイトに設定した．結果は**表 1** のようになった．AProVE Web Interface 版を用いた証明では図 5 および図 8 のプログラムの停止性を証明することができた．それ以外のプログラムは最大タイムアウト時間である 120 秒を超えても証明できなかった．μ-Term Web Interface 版を用いた証明ではすべてのプログラムについて，最大タイムアウト時間である 5 秒を超えても証明できなかった．タイムアウト時間の制限がないダウンロード版 AProVE1.2 ではいずれのプログラムに対しても 12 時間計算したが証明は完了せず，停止性の証明はできなかった．さらに例外発生の条件をより複雑にした 10 個程度のプログラムの証明を試みたが，例外処理を含むプログラムで停止性または非停止性を証明できたのは図 5 のみであった．

⋆1 http://aprove.informatik.rwth-aachen.de/
⋆2 http://zenon.dsic.upv.es/muterm/

表 1　停止性・非停止性証明結果
Table 1　Result of (non-)termination proof.

program	tool	result	time
Fig. 2	AProVE(Web)	time out	$>$ 120 sec.
(with termination)	μ-Term(Web)	time out	$>$ 5 sec.
	AProVE(1.2)	time out	$>$ 12 hours
Fig. 5	AProVE(Web)	termination	2.48 sec.
(with termination)	μ-Term(Web)	time out	$>$ 5 sec.
	AProVE(1.2)	time out	$>$ 12 hours
Fig. 6	AProVE(Web)	time out	$>$ 120 sec.
(with non-termination)	μ-Term(Web)	time out	$>$ 5 sec.
	AProVE(1.2)	time out	$>$ 12 hours
Fig. 7	AProVE(Web)	time out	$>$ 120 sec.
(with termination)	μ-Term(Web)	time out	$>$ 5 sec.
	AProVE(1.2)	time out	$>$ 12 hours
Fig. 8	AProVE(Web)	termination	52.14 sec.
(with termination)	μ-Term(Web)	time out	$>$ 5 sec.
(not including exception)	AProVE(1.2)	time out	$>$ 12 hours

現在のところ最内評価戦略と文脈依存評価を併用した場合には CS-TRS の停止性証明ツールは，それらを併用しない場合と比較して非力である．実験結果のように，多くの場合で AProVE や μ-Term では証明ができなかった．このため強力な CS-TRS の最内停止性証明ツールの開発が必要である．

6. おわりに

我々は例外処理を持つ関数型プログラムの停止性・非停止性を証明する手法を提案した．本手法では例外処理を持つ関数型プログラムを文脈依存項書き換え系（CS-TRS）に変換し，

CS-TRS の停止性証明ツールを用いて証明する．本手法による変換は停止性に関する健全性と完全性が成立することを証明し，CS-TRS の停止性証明ツールを用いてプログラムの停止性・非停止性が証明可能であることを示した．

しかし，既存の停止性証明ツールは最内評価戦略と文脈依存戦略を併用した場合に非力であり，多くのプログラムについて停止性の判定ができない．本手法の有効性を高めるためには，この問題を解決する必要があるが，これには 2 つのアプローチが考えられる．

- より多くの CS-TRS について停止性証明が可能な強力なツールの開発．
- 既存のツールで証明できる CS-TRS を出力するように変換法を変更する．

2 番目のアプローチについては，たとえば完全性が成り立たなくても既存のツールで証明できることを優先した変換法の検討が考えられる．完全性が成り立たない場合，変換後の CS-TRS が停止性を持たないことが分かってもプログラムが停止性を持たないとは限らないが，健全性が成り立てば CS-TRS の停止性が証明できればプログラムの停止性は保証される．また，termination graph を用いた関数型プログラムの停止性判定法[7] を例外処理を持つ関数型プログラムに適用できるように拡張する手法も考えられる．

また，本論文で対象としたプログラミング言語 ML/ex は機能が限られているが，これを拡張することも今後の課題としてあげることができる．ML/ex は変数と値の結び付きを保持する環境を持っていないが，Standard ML 等多くの関数型言語では `let` 式を使って環境を持つことができる．ML/ex に環境を持たせるような拡張が必要である．また，ML/ex は先行評価に基づく言語であるが，遅延評価に基づくプログラミングを対象とすることも今後の課題である．

参 考 文 献

1) 住井英二郎，大根田裕一，米澤明憲：例外処理機構を備えた命令型言語の CPS 変換とその定式化，情報処理学会論文誌 プログラミング，Vol.45, No.SIG 12 (PRO 23).
2) Benton, N. and Kennedy, A.: Exception Syntax, *Journal of Functional Programming*, Vol.4, No.11, pp.395–410 (2001).
3) Schröder, L. and Mossakowski, T.: Generic Exception Handling and the Java Monad, *Proc. Algebraic Methodology and Software Technology 10th International Conference AMAST 2004*, Lecture Notes in Computer Science 3116, Stirling, UK, pp.443–459 (2004).
4) 馬場正貴，酒井正彦，濱口　毅，西田直樹，坂部俊樹，草刈圭一朗：例外処理を持つ関数型プログラムの停止性証明法，第 12 回プログラミングおよびプログラミング言語ワークショップ PPL2010 論文集，p.81 (2010).
5) Giesl, J., Thiemann, R., Schneider-Kamp, P. and Falke, S.: Automated Termination Proofs with AProVE, *Proc. 15th International Conference on Rewriting Techniques and Applications* (*RTA-04*), Lecture Notes in Computer Science 3091, Aachen, Germany, pp.210–220 (2004).
6) Alarcón, B., Gutiérrez, R., Iborra, J. and Lucas, S.: Proving termination of context-sensitive rewriting with MU-TERM, *Proc. 6th Spanish Conference on Programming and Computer Languages, PROLE 2006* (*Electronic Notes in Theoretical Computer Science, 188*), pp.105–115 (2007).
7) Giesl, J., Swiderski, S., Schneider-Kamp, P. and Thiemann, R.: Automated Termination Analysis for Haskell: From Term Rewriting to Programming Languages, *Proc. Term Rewriting and Applications 17th International Conference RTA 2006*, Lecture Notes in Computer Science 4098, Seattle, USA, pp.297–312 (2006).

付　　録

以下では特に断らない限り，*Fundef* に根重なりがないプログラム $\mathcal{P}$ を変換した CS-TRS $\phi(\mathcal{P})$ を対象とする．$\phi(\mathcal{P})$ による最内書き換え $\underset{in}{\rightarrow}_{\phi(\mathcal{P})}$ を $\rightarrow$ と略記する．CS-TRS $\phi(\mathcal{P})$ の項のうち，$\langle \mathcal{R}, \mu \rangle$ 最内停止性を持つものの集合を *ISN* とする．また，最内正規形の集合を *NF* とする．

A.1　定理 4.2（プログラムの評価と書き換えの一致）の証明

補題 A.1.1（最内リデックスの一意性）　項の $\phi(\mathcal{P})$ による最内書き換えのリデックスは存在するならば一意である．

証明　$\phi(\mathcal{P})$ のどの関数 f についても $|\mu(f)| \leq 1$ であるため，最内書き換えのリデックスは一意である．□

補題 A.1.2　$t \rightarrow s_1$ かつ $t \rightarrow s_2$ ならば $s_1 = s_2$

証明　補題 A.1.1 より t のリデックスは唯一に定まり，また $\phi(\mathcal{P})$ には根重なりがないため適用できる規則は一意に定まるため，題意は成立する．□

補題 A.1.3（正規形の一意性）　いかなる項についてもその最内正規形は一意である．

証明　補題 A.1.2 より明らか．□

任意の項 t について，その最内正規形が唯一に定まるとき，それを $t{\Downarrow}$ で表す．

定義 A.1.4（項のサイズ）　項 t のサイズ $|t|$ を $|t| = |\mathcal{P}os(t)|$ と定義する．また，項のリスト $[t_1, \ldots, t_n]$ のサイズ $|[t_1, \ldots, t_n]|$ を $|[t_1, \ldots, t_n]| = |t_1| + \cdots + |t_n|$ と定義する．

補題 A.1.5　項 t を $\mathcal{T}(\mathcal{F}, \mathcal{V})$ 上の項，代入 θ を $\mathcal{V}ar(t) \subseteq \mathcal{D}om(\theta)$，$\theta : \mathcal{V} \rightarrow \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ とする．このとき以下の (1) から (6) が成り立つ．

(1)　$ev(t,\theta) = v\downarrow$ ならば $t\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$

(2)　$ev(t,\theta) = E\uparrow$ ならば $t\theta \to^* \mathit{fire}(E)$

(3)　$ev(t,\theta) = \bot$ ならば $t\theta \to^* b \in NF \cap (\mathcal{T}(\mathcal{F}')\backslash(\mathcal{T}(\mathcal{F}_\mathcal{C}) \cup \{\mathit{fire}(E) \mid E \in \mathit{Except}\}))$

(4)　$\mathit{evArgs}([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ ならば各 i ($1 \le i \le n$) について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$

(5)　$\mathit{evArgs}([t_1,\ldots,t_n],\theta) = E\uparrow$ とする．このとき，ある $1 \le i \le n$ について，$t_i\theta \to^* \mathit{fire}(E)$，かつ $1 \le j < i$ であるすべての j について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$

(6)　$\mathit{evArgs}([t_1,\ldots,t_n],\theta) = \bot$ とする．このとき，ある $1 \le i \le n$ について $t_i\theta \to^* b_i \in NF \cap (\mathcal{T}(\mathcal{F}')\backslash(\mathcal{T}(\mathcal{F}_\mathcal{C}) \cup \{\mathit{fire}(E) \mid E \in \mathit{Except}\}))$ かつ，$1 \le j < i$ であるすべての j について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$

証明　(1) から (6) を関数 ev と関数 evArgs の定義に関する同時帰納法，すなわち項の値が求まるまで定義 3.3 と定義 3.4 を使用した回数に関する帰納法で証明する．

(1)　$ev(t,\theta) = v\downarrow$ とする．このとき，以下の場合が存在する．

- $t = x \in \mathcal{V}$ のとき，$t\theta = x\theta = v$ より $t\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$.
- $t = \mathit{succ}(t')$ のとき，$ev(t',\theta)$ の値は $v'\downarrow$ であり，かつ $\mathit{succ}(v') = v$ である．帰納法の仮定 (1) より $t'\theta \to^* v' \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であるから，
$$t\theta = \mathit{succ}(t'\theta) \to^* \mathit{succ}(v') = v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$$
- $t = c \in (\mathcal{F}_\mathcal{C}\backslash\{\mathit{succ}\})$ のとき，$v = c$. また，$t\theta = c = v$ であるから $t\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$.
- $t = \mathit{handle}(t_1,t_2,t_2)$ のとき，$ev(t,\theta) = v\downarrow$ となるのは $t_2 = E \in \mathit{Except}$ であり，なおかつ $ev(t_1,\theta)$ の値によって以下の 2 つの場合がある．
  - $ev(t_1,\theta) = v\downarrow$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であるから，
$$\begin{aligned} t\theta &= \mathit{handle}(t_1\theta,E,t_3\theta)\\ &\to^* \mathit{handle}(v,E,t_3\theta)\\ &\to \mathit{select}(\mathit{isData}(v),v,E,t_3\theta)\\ &\to^* \mathit{select}(\mathit{tt},v,E,t_3\theta)\\ &\to v \in \mathcal{T}(\mathcal{F}_\mathcal{C}) \end{aligned}$$
  - $ev(t_1,\theta) = E\uparrow$ かつ $ev(t_3,\theta) = v\downarrow$ のとき，帰納法の仮定 (2) より $t_1\theta \to^* \mathit{fire}(E)$ であり，帰納法の仮定 (1) より $t_3\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であるから，
$$\begin{aligned} t\theta &= \mathit{handle}(t_1\theta,E,t_3\theta)\\ &\to^* \mathit{handle}(\mathit{fire}(E),E,t_3\theta)\\ &\to \mathit{select}(\mathit{isData}(\mathit{fire}(E)),\mathit{fire}(E),E,t_3\theta)\\ &\to \mathit{select}(\mathit{fire}(E),\mathit{fire}(E),E,t_3\theta)\\ &\to t_3\theta\\ &\to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C}) \end{aligned}$$
- $t = \mathit{if}(t_1,t_2,t_3)$ のとき，$ev(t,\theta) = v\downarrow$ となるのは $ev(t_1,\theta)$ の値により，次の 2 つの場合がある．
  - $ev(t_1,\theta) = \mathit{True}\downarrow$ かつ $ev(t_2,\theta) = v\downarrow$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* \mathit{True}$ ならびに $t_2\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であるから，
$$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(\mathit{True},t_2\theta,t_3\theta)\\ &\to t_2\theta\\ &\to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C}) \end{aligned}$$
  - $ev(t_1,\theta) = \mathit{False}\downarrow$ かつ $ev(t_3,\theta) = v\downarrow$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* \mathit{False}$ ならびに $t_3\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であるから，
$$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(\mathit{False},t_2\theta,t_3\theta)\\ &\to t_3\theta\\ &\to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C}) \end{aligned}$$
- $t = f(t_1,\ldots,t_n)$ かつ $f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_\mathcal{D}$ のとき，$\mathit{evArgs}([t_1,\ldots,t_n],\theta)$ の値は $[v_1,\ldots,v_n]\downarrow$ であり，$f(l_1,\ldots,l_n) \to r \in \mathit{Fundef} \cup \mathit{BuiltinDef}$ かつ $\sigma = \mathit{match}[v_1,\ldots,v_n][l_1,\ldots,l_n]$，$ev(r,\sigma) = v\downarrow$ である．帰納法の仮定 (4) から各 i ($1 \le i \le n$) について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ より，
$$\begin{aligned} t\theta &= f(t_1,t_2,\ldots,t_n)\theta\\ &= f(t_1\theta,t_2\theta,\ldots,t_n\theta)\\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta)\\ &\to^* f_1(v_1,t_2\theta,\ldots,t_n\theta)\\ &\to \mathit{guard}(\mathit{isData}(v_1),f_2(v_1,t_2\theta,\ldots,t_n\theta))\\ &\to^* \mathit{guard}(\mathit{tt},f_2(v_1,t_2\theta,\ldots,t_n\theta))\\ &\to f_2(v_1,t_2\theta,\ldots,t_n\theta) \end{aligned}$$

$\to^* \cdots$
$\to^* f_{n+1}(v_1, \ldots, v_n)$
$= f_{n+1}(l_1\sigma, \ldots, l_n\sigma)$
$\to r\sigma$

また帰納法の仮定 (1) から $r\sigma \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ より，$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_\mathcal{C})$.

(2)　$ev(t, \theta) = E\uparrow$ であるとする．このとき，以下の場合が存在する．

- $t = succ(t')$ のとき，$ev(t', \theta) = E\uparrow$ である．帰納法の仮定 (2) より $t'\theta \to^* fire(E)$ であるから，
  $t\theta = succ(t'\theta) \to^* succ(fire(E)) \to fire(E)$
- $t = raise(t')$ のとき，$ev(t, \theta) = E\uparrow$ となるのは $t' = E \in Except$ のときである．よって，$t\theta = raise(E) \to fire(E)$.
- $t = handle(t_1, t_2, t_3)$ のとき，$ev(t, \theta) = E\uparrow$ となるのは $t_2 = E' \in Except$ であり，なおかつ $ev(t_1, \theta)$ の値によって以下の 2 つの場合がある．
  - $ev(t_1, \theta) = E'\uparrow$ かつ $ev(t_3, \theta) = E\uparrow$ のとき，帰納法の仮定 (2) より $t_1\theta \to^* fire(E')$，ならびに $t_3\theta \to^* fire(E)$ であるから，
    $t\theta = handle(t_1\theta, E', t_3\theta)$
    $\to^* handle(fire(E'), E', t_3\theta)$
    $\to select(isData(fire(E')), fire(E'), E', t_3\theta)$
    $\to select(fire(E'), fire(E'), E', t_3\theta)$
    $\to t_3\theta$
    $\to^* fire(E)$
  - $E \neq E'$ かつ $ev(t_1, \theta) = E\uparrow$ のとき，帰納法の仮定 (2) より $t_1\theta \to^* fire(E)$ であるから，
    $t\theta = handle(t_1\theta, E', t_3\theta)$
    $\to^* handle(fire(E), E', t_3\theta)$
    $\to select(isData(fire(E)), fire(E), E', t_3\theta)$
    $\to select(fire(E), fire(E), E', t_3\theta)$
    $\to fire(E)$
- $t = if(t_1, t_2, t_3)$ のとき $ev(t, \theta) = E\uparrow$ となるのは $ev(t_1, \theta)$ の値により，以下の 3 つの場合がある．
  - $ev(t_1, \theta) = True\downarrow$ かつ $ev(t_2, \theta) = E\uparrow$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* True$ であり，帰納法の仮定 (2) より $t_2\theta \to^* fire(E)$ であるから，
    $t\theta = if(t_1\theta, t_2\theta, t_3\theta)$
    $\to^* if(True, t_2\theta, t_3\theta)$
    $\to t_2\theta$
    $\to^* fire(E)$
  - $ev(t_1, \theta) = False\downarrow$ かつ $ev(t_3, \theta) = E\uparrow$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* False$ であり，帰納法の仮定 (2) より $t_3\theta \to^* fire(E)$ であるから，
    $t\theta = if(t_1\theta, t_2\theta, t_3\theta)$
    $\to^* if(False, t_2\theta, t_3\theta)$
    $\to t_3\theta$
    $\to^* fire(E)$
  - $ev(t_1, \theta) = E\uparrow$ のとき帰納法の仮定 (2) より $t_1\theta \to^* fire(E)$ であるから，
    $t\theta = if(t_1\theta, t_2\theta, t_3\theta)$
    $\to^* if(fire(E), t_2\theta, t_3\theta)$
    $\to fire(E)$
- $t = f(t_1, \ldots, t_n)$，$f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_\mathcal{D}$ のとき，$ev(t, \theta) = E\uparrow$ となるのは $ev([t_1, \ldots, t_n], \theta)$ の値により以下の 2 つの場合がある．
  - $evArgs([t_1, \ldots, t_n], \theta) = [v_1, \ldots, v_n]\downarrow$ のとき，$f(l_1, \ldots, l_n) \to r \in Fundef \cup BuiltinDef$，$\sigma = match[v_1, \ldots, v_n][l_1, \ldots, l_n]$ かつ $ev(r, \sigma) = E\uparrow$ である．帰納法の仮定 (4) から各 i $(1 \leq i \leq n)$ について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ より，
    $t\theta = f(t_1\theta, \ldots, t_n\theta)$
    $\to f_1(t_1\theta, t_2\theta, \ldots, t_n\theta)$
    $\to^* f_1(v_1, t_2\theta, \ldots, t_n\theta)$
    $\to guard(isData(v_1), f_2(v_1, t_2\theta, \ldots, t_n\theta))$
    $\to^* guard(tt, f_2(v_1, t_2\theta, \ldots, t_n\theta))$
    $\to f_2(v_1, t_2\theta, \ldots, t_n\theta)$
    $\to^* \cdots$
    $\to^* f_{n+1}(v_1, \ldots, v_n)$
    $= f_{n+1}(l_1\sigma, \ldots, l_n\sigma)$
    $\to r\sigma$

また，帰納法の仮定 (2) より $r\sigma \to^* \mathit{fire}(E)$ であるから，$t\theta \to^* \mathit{fire}(E)$.

– $\mathit{evArgs}([t_1,\ldots,t_n],\theta) = E\uparrow$ のとき，帰納法の仮定 (5) より，ある i（$1 \le i \le n$）について $t_i\theta \to^* \mathit{fire}(E)$，かつすべての j（$1 \le j < i$）について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ である．よって，

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta)\\ &\to f_1(t_1\theta,\ldots,t_n\theta)\\ &\to^* f_i(v_1,\ldots,v_{i-1},t_i\theta,t_{i+1}\theta,\ldots,t_n\theta)\\ &\to^* f_i(v_1,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta)\\ &\to \mathit{guard}(\mathit{isData}(\mathit{fire}(E)),f_{i+1}(v_1,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta))\\ &\to \mathit{guard}(\mathit{fire}(E),f_{i+1}(v_1,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta))\\ &\to \mathit{fire}(E) \end{aligned}$$

したがって，$t\theta \to^* \mathit{fire}(E)$.

(3)　$\mathit{ev}(t,\theta) = \bot$ とする．このとき，以下の場合がある．ここで，$\mathcal{T}^{\mathcal{NC}}(\mathcal{F}') = NF \cap (\mathcal{T}(\mathcal{F}')\backslash(\mathcal{T}(\mathcal{F}_\mathcal{C}) \cup \{\mathit{fire}(E) \mid E \in \mathit{Except}\}))$ とおく．

- $t = E \in \mathit{Except}$ のとき，$\mathit{ev}(E,\theta) = \bot$ であり，$E\theta = E \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$.
- $t = \mathit{succ}(t')$ のとき，$\mathit{ev}(t',\theta) = \bot$ である．帰納法の仮定 (3) より $t'\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ であるから，

  $t\theta = \mathit{succ}(t'\theta) \to^* \mathit{succ}(b)$

  ここで，$\mathit{succ}(b) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ である．よって (3) を満たす.
- $t = \mathit{raise}(t')$ のとき，$\mathit{ev}(t,\theta) = \bot$ となるのは $t' \notin \mathit{Except}$ のときである．このとき $t\theta = \mathit{raise}(t'\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ である．
- $t = \mathit{handle}(t_1,t_2,t_3)$ のとき，$\mathit{ev}(t,\theta) = \bot$ となるのは以下の 3 つの場合がある．
  – $t_2 = E \in \mathit{Except}$ かつ $\mathit{ev}(t_1,\theta) = E\uparrow$ かつ $\mathit{ev}(t_3,\theta) = \bot$ のとき，帰納法の仮定 (2) より $t_1\theta \to^* \mathit{fire}(E)$ であり，帰納法の仮定 (3) より $t_3\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$.

  $$\begin{aligned} t\theta &= \mathit{handle}(t_1\theta,E,t_3\theta)\\ &\to^* \mathit{handle}(\mathit{fire}(E),E,t_3\theta)\\ &\to \mathit{select}(\mathit{isData}(\mathit{fire}(E)),\mathit{fire}(E),E,t_3\theta)\\ &\to \mathit{select}(\mathit{fire}(E),\mathit{fire}(E),E,t_3\theta)\\ &\to t_3\theta\\ &\to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

  – $t_2 = E \in \mathit{Except}$ かつ $\mathit{ev}(t_1,\theta) = \bot$ のとき，帰納法の仮定 (3) より $t_1\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$.

  $$\begin{aligned} t\theta &= \mathit{handle}(t_1\theta,E,t_3\theta)\\ &\to^* \mathit{handle}(b,E,t_3\theta)\\ &\to \mathit{select}(\mathit{isData}(b),b,E,t_3\theta) \end{aligned}$$

  ここで，b により以下の 2 つの場合がある．
  * $\mathit{root}(b) = \mathit{succ}$ のとき，$b = \mathit{succ}^k(b')$（$k > 1$）かつ $\mathit{root}(b') \ne \mathit{succ}$ とする．このとき，$b' \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ である．$\mathit{select}(\mathit{isData}(b),b,E,t_3\theta) \to^* \mathit{select}(\mathit{isData}(b'),b,E,t_3\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$
  * $\mathit{root}(b) \ne \mathit{succ}$ のとき，$\mathit{select}(\mathit{isData}(b),b,E,t_3\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$

  – $t_2 \notin \mathit{Except}$ のとき，$t\theta = \mathit{handle}(t_1\theta,t_2\theta,t_3\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$.
- $t = \mathit{if}(t_1,t_2,t_3)$ のとき $\mathit{ev}(t,\theta) = \bot$ となるのは $\mathit{ev}(b,\theta)$ の値により以下の 4 つの場合がある．
  – $\mathit{ev}(t_1,\theta) = \mathit{True}\downarrow$ かつ $\mathit{ev}(t_2,\theta) = \bot$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* \mathit{True}$ であり，帰納法の仮定 (3) より $t_2\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ であるから，

  $$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(\mathit{True},t_2\theta,t_3\theta)\\ &\to t_2\theta\\ &\to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

  – $\mathit{ev}(t_1,\theta) = \mathit{False}\downarrow$ かつ $\mathit{ev}(t_3,\theta) = \bot$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* \mathit{False}$ であり，帰納法の仮定 (3) より $t_3\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ であるから，

  $$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(\mathit{False},t_2\theta,t_3\theta)\\ &\to t_3\theta\\ &\to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

  – $\mathit{ev}(t_1,\theta) = t_1'\downarrow$ かつ $t_1' \ne \mathit{True}$ かつ $t_1' \ne \mathit{False}$ のとき，帰納法の仮定 (1) より $t_1\theta \to^* t_1' \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であり，

  $$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(t_1',t_2\theta,t_3\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

– $ev(t_1,\theta)=\bot$ のとき帰納法の仮定 (3) より $t_1\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$

$$\begin{aligned} t\theta &= \mathit{if}(t_1\theta,t_2\theta,t_3\theta)\\ &\to^* \mathit{if}(b,t_2\theta,t_3\theta) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

- $t = f(t_1,\ldots,t_n)$, $f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_{\mathcal{D}}$ のとき，$ev(t,\theta)=\bot$ となるのは $ev([t_1,\ldots,t_n],\theta)$ の値により以下の 2 つの場合がある．
  – $evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ のとき，$f(l_1,\ldots,l_n) \to r \in \mathit{Fundef} \cup \mathit{BuiltinDef}$, $\sigma = \mathit{match}[v_1,\ldots,v_n][l_1,\ldots,l_n]$ かつ $ev(r,\sigma)=\bot$ である．帰納法の仮定 (4) から各 i ($1 \le i \le n$) について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F_C})$ より，

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta)\\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta)\\ &\to^* f_1(v_1,t_2\theta,\ldots,t_n\theta)\\ &\to \mathit{guard}(\mathit{isData}(v_1), f_2(v_1,t_2\theta,\ldots,t_n\theta))\\ &\to^* \mathit{guard}(\mathit{tt}, f_2(v_1,t_2\theta,\ldots,t_n\theta))\\ &\to f_2(v_1,t_2\theta,\ldots,t_n\theta)\\ &\to^* \cdots\\ &\to^* f_{n+1}(v_1,\ldots,v_n)\\ &= f_{n+1}(l_1\sigma,\ldots,l_n\sigma)\\ &\to r\sigma \end{aligned}$$

また，帰納法の仮定 (3) より $r\sigma \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ であるから，$t\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$.

  – $evArgs([t_1,\ldots,t_n],\theta)=\bot$ のとき，帰納法の仮定 (6) より，ある i ($1 \le i \le n$) について，$t_i\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$，かつ，すべての j ($1 \le j < i$) について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F_C})$ である．よって，

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta)\\ &\to f_1(t_1\theta,\ldots,t_n\theta)\\ &\to^* f_i(v_1,\ldots,v_{i-1},t_i\theta,t_{i+1}\theta,\ldots,t_n\theta)\\ &\to^* f_i(v_1,\ldots,v_{i-1},b,t_{i+1}\theta,\ldots,t_n\theta)\\ &\to \mathit{guard}(\mathit{isData}(b), f_{i+1}(v_1,\ldots,v_{i-1},b,t_{i+1}\theta,\ldots,t_n\theta)) \end{aligned}$$

ここで b により，以下の 2 つの場合がある．
    * $root(b) = succ$ のとき，$b = succ^k(b')$ ($k>1$) かつ $root(b') \neq succ$ とする．このとき，$b' \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ である．

$$\begin{aligned} &\mathit{guard}(\mathit{isData}(b), f_{i+1}(v_1,\ldots,v_{i-1},b,t_{i+1}\theta,\ldots,t_n\theta))\\ \to^* &\mathit{guard}(\mathit{isData}(b'), f_{i+1}(v_1,\ldots,v_{i-1},b,t_{i+1}\theta,\ldots,t_n\theta))\\ \in &\mathcal{T}^{\mathcal{NC}}(\mathcal{F}') \end{aligned}$$

    * $root(b) \neq succ$ のとき，
    $\mathit{guard}(\mathit{isData}(b), f_{i+1}(v_1,\ldots,v_{i-1},b,t_{i+1}\theta,\ldots,t_n\theta)) \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$

(4) $evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ ($0 \le n$) とする．
- $n=0$ のとき，$evArgs([\,],\theta)=[\,]$ より明らか．
- $n>0$ のとき，$ev(t_1,\theta)=v_1\downarrow$ かつ $evArgs([t_2,\ldots,t_n],\theta)=[v_2,\ldots,v_n]\downarrow$ である．帰納法の仮定 (1) より $t_1\theta \to^* v_1 \in \mathcal{T}(\mathcal{F_C})$ であり，帰納法の仮定 (4) から各 j ($2 \le j \le n$) について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F_C})$ であるから (4) を満たす．

(5) $evArgs([t_1,\ldots,t_n],\theta) = E\uparrow$ とする．このとき，$n \neq 0$ であり，$ev(t_1,\theta)$ の値により以下の 2 つの場合がある．
- $ev(t_1,\theta) = v_1\downarrow$ のとき，$evArgs([t_2,\ldots,t_n],\theta) = E\uparrow$ である．帰納法の仮定 (1) から $t_1\theta \to^* v_1 \in \mathcal{T}(\mathcal{F_C})$ である．また，帰納法の仮定 (5) からある i ($2 \le i \le n$) について $t_i\theta \to^* \mathit{fire}(E)$ かつ $2 \le j < i$ なるすべての j について，$t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F_C})$．したがって (5) を満たす．
- $ev(t_1,\theta) = E\uparrow$ のとき，帰納法の仮定 (2) より $t_1\theta \to^* \mathit{fire}(E)$ であり，(5) を満たす．

(6) $evArgs([t_1,\ldots,t_n],\theta) = \bot$ とする．このとき，$n \neq 0$ であり，$ev(t_1,\theta)$ の値により以下の 2 つの場合がある．
- $ev(t_1,\theta) = v_1\downarrow$ のとき，$evArgs([t_2,\ldots,t_n],\theta) = \bot$ である．帰納法の仮定 (1) から $t_1\theta \to^* v_1 \in \mathcal{T}(\mathcal{F_C})$ である．また，帰納法の仮定 (6) からある i ($2 \le i \le n$) について $t_i\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ かつ $2 \le j < i$ なるすべての j について，$t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F_C})$．したがって (6) を満たす．
- $ev(t_1,\theta) = \bot$ のとき，帰納法の仮定 (3) より $t_1\theta \to^* b \in \mathcal{T}^{\mathcal{NC}}(\mathcal{F}')$ であり，(6) を満たす．

□

補題 A.1.6　項 t を $\mathcal{T}(\mathcal{F},\mathcal{V})$ 上の項，代入 θ を $\mathcal{V}ar(t) \subseteq \mathcal{D}om(\theta)$，$\theta : \mathcal{V} \to \mathcal{T}(\mathcal{F_C})$ とする．このとき以下の (1) から (4) が成り立つ．

(1) $t\theta \to^* v \in \mathcal{T}(\mathcal{F_C})$ ならば $ev(t,\theta) = v\downarrow$

(2)　$t\theta \to^* \mathit{fire}(E)$ ならば $ev(t,\theta) = E\uparrow$

(3)　任意の $f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_{\mathcal{D}}$ について $f(t_1,\ldots,t_n)\theta \to^* f_{n+1}(v_1,\ldots,v_n)$ かつ各 i（$1 \le i \le n$）について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ ならば $evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$

(4)　任意の $f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_{\mathcal{D}}$ についてある i（$1 \le i \le n$）が存在して $f(t_1,\ldots,t_n)\theta \to^* f_i(v_1,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta)$ かつ，$1 \le j < i$ であるすべての j について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ ならば $evArgs([t_1,\ldots,t_n],\theta) = E\uparrow$

証明　(1) から (4) をそれぞれに現れる書き換えの総ステップ数と項のサイズに関する帰納法で証明する．すなわち総ステップ数が k で (1) から (4) が成り立つことを仮定して総ステップ数が l（$k < l$）のとき成り立つことを証明する．総ステップ数が同じときは項のサイズが m のとき (1) から (4) が成り立つことを仮定して項のサイズが n（$m < n$）のとき成り立つことを証明する．なお，総ステップ数が同じで項のサイズを利用するのは (1) の $t = succ(t')$ のときのみであり，その他は総ステップ数の帰納法だけで証明している．ここで補題 A.1.2 よりその書き換え系列は一意に定まる．

(1)　$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ であるとする．CS-TRS の定義から $t\theta$ が $v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ に書き換えられるのは以下の場合のみである．

- $t \in \mathcal{V}$ のとき，$t\theta = v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ であり，$ev(t,\theta) = t\theta = v\downarrow$.
- $t = succ(t')$ のとき，$t\theta = succ(t'\theta) \to^* succ(v') = v$ となる．$v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ より，$t'\theta \to^* v' \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$．$t'\theta \to^* v' \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ と $t\theta \to^* v$ の書き換えの総ステップ数は同じだが $|t| > |t'|$ なので，帰納法の仮定 (1) から $ev(t',\theta) = v'\downarrow$．したがって $ev(t,\theta) = succ(v')\downarrow$.
- $t = c \in (\mathcal{F}_{\mathcal{C}} \backslash \{succ\})$ のとき，$t\theta = c = v$ より，$ev(t,\theta) = c\downarrow = v\downarrow$
- $t = handle(t_1,t_2,t_3)$ のとき，$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ となるためには $t_2 = E \in Except$ でなければならない．$t_1\theta\Downarrow = t_1' \in NF$ とすると，$t\theta$ は以下のように書き換えられる．

$$\begin{aligned} t\theta &= handle(t_1\theta, E, t_3\theta) \\ &\to^* handle(t_1', E, t_3\theta) \\ &\to select(isData(t_1'), t_1', E, t_3\theta) \end{aligned}$$

$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ となるのは $isData(t_1')$ の書き換え結果によって以下の 2 つの場合がある．

– $isData(t_1') \to^* tt$ のとき，

$$\begin{aligned} select(isData(t_1'), t_1', E, t_3\theta) &\to^* select(tt, t_1', E, t_3\theta) \\ &\to t_1' \end{aligned}$$

$t_1' \in NF$ より，$t_1' = v$ である．$t_1\theta \to^* v$ であるから帰納法の仮定 (1) より $ev(t_1,\theta) = v\downarrow$．したがって $ev(t,\theta) = v\downarrow$.

– $isData(t_1') \to^* \mathit{fire}(E)$ のとき，$t_1' = \mathit{fire}(E)$ であり，

$$\begin{aligned} & select(isData(t_1'), t_1', E, t_3\theta) \\ =\ & select(isData(\mathit{fire}(E)), \mathit{fire}(E), E, t_3\theta) \\ \to\ & select(\mathit{fire}(E), \mathit{fire}(E), E, t_3\theta) \\ \to\ & t_3\theta \\ \to^*\ & v \end{aligned}$$

$t_1\theta \to^* \mathit{fire}(E)$ なので，帰納法の仮定 (2) より $ev(t_1,\theta) = E\uparrow$．よって，$ev(t,\theta) = ev(t_3,\theta)$．帰納法の仮定 (1) より $ev(t_3,\theta) = v\downarrow$．したがって，$ev(t,\theta) = v\downarrow$.

- $t = if(t_1,t_2,t_3)$ のとき，$t\theta = if(t_1\theta, t_2\theta, t_3\theta)$ である．$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ となるのは以下の 2 つの場合がある．
  – $t_1\theta \to^* True$ のとき，$t\theta \to^* if(True, t_2\theta, t_3\theta) \to t_2\theta \to^* v$ である．帰納法の仮定 (1) より $ev(t_1,\theta) = True\downarrow$ かつ $ev(t_2,\theta) = v\downarrow$ であるため $ev(t,\theta) = v\downarrow$.
  – $t_1\theta \to^* False$ のとき，$t\theta \to^* if(False, t_2\theta, t_3\theta) \to t_3\theta \to^* v$ である．帰納法の仮定 (1) より $ev(t_1,\theta) = False\downarrow$ かつ $ev(t_3,\theta) = v\downarrow$ であるため，$ev(t,\theta) = v\downarrow$.
- $t = f(t_1,\ldots,t_n)$，$f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_{\mathcal{D}}$ のとき，$t\theta \to^* v \in \mathcal{T}(\mathcal{F}_{\mathcal{C}})$ となるのは以下のように $f_1, f_2, \ldots, f_{n+1}$ を経由して書き換えられる場合のみである．

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta, t_2\theta, \ldots, t_n\theta) \\ &\to^* f_1(v_1, t_2\theta, \ldots, t_n\theta) \\ &\to guard(isData(v_1), f_2(v_1, t_2\theta, \ldots, t_n\theta)) \\ &\to guard(tt, f_2(v_1, t_2\theta, \ldots, t_n\theta)) \\ &\to f_2(v_1, t_2\theta, \ldots, t_n\theta) \\ &\to^* \cdots \\ &\to^* f_{n+1}(v_1,\ldots,v_n) \end{aligned}$$

さらに $f(l_1,\ldots,l_n) \to r \in \mathit{Fundef} \cup \mathit{BuiltinDef}$ と σ が存在し，$\sigma = \mathit{match}[v_1,\ldots,v_n][l_1,\ldots,l_n]$ かつ $f_{n+1}(v_1,\ldots,v_n) \to r\sigma$ なので，

$$\begin{aligned} f_{n+1}(v_1,\ldots,v_n) &\to r\sigma \\ &\to^* v \end{aligned}$$

ここで，$\to$ は最内書き換えであるため $1 \le i \le n$ であるすべての i について $v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$．よって $f(t_1,\ldots,t_n)\theta \to^* f_{n+1}(v_1,\ldots,v_n)$ なので，帰納法の仮定 (3) より，$\mathit{evArgs}([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]{\downarrow}$ である．$r\sigma \to^* v$ であるから帰納法の仮定 (1) より $\mathit{ev}(r,\sigma) = v{\downarrow}$．よって，$\mathit{ev}(t,\theta) = \mathit{ev}(r,\sigma) = v{\downarrow}$ である．

(2)　$t\theta \to^* \mathit{fire}(E)$ であるとする．CS-TRS の定義から $t\theta$ が $\mathit{fire}(E)$ に書き換えられるのは以下の場合のみである．

- $t = \mathit{succ}(t')$ のとき，$t\theta = \mathit{succ}(t'\theta) \to^* \mathit{succ}(\mathit{fire}(E)) \to \mathit{fire}(E)$ となる．$t'\theta \to^* \mathit{fire}(E)$ なので，帰納法の仮定 (2) より $\mathit{ev}(t',\theta) = E{\uparrow}$．したがって $\mathit{ev}(t,\theta) = E{\uparrow}$．
- $t = \mathit{raise}(t')$ のとき，$t\theta = \mathit{raise}(t'\theta) \to^* \mathit{fire}(E)$ より，$t' = E$．よって $t = \mathit{raise}(E)$ であり，$\mathit{ev}(t,\theta) = E{\uparrow}$
- $t = \mathit{handle}(t_1,t_2,t_3)$ のとき，$t\theta \to^* \mathit{fire}(E)$ となるためには $t_2 = E' \in \mathit{Except}$ でなければならない．$t_1\theta{\Downarrow} = t_1' \in \mathit{NF}$ とすると，$t\theta$ は以下のように書き換えられる．

$$\begin{aligned} t\theta &= \mathit{handle}(t_1\theta, E', t_3\theta) \\ &\to^* \mathit{handle}(t_1', E', t_3\theta) \\ &\to \mathit{select}(\mathit{isData}(t_1'), t_1', E', t_3\theta) \end{aligned}$$

$t\theta \to^* \mathit{fire}(E)$ となるのは $\mathit{isData}(t_1')$ の書き換え結果によって以下の 2 つの場合がある．

- $\mathit{isData}(t_1') \to \mathit{fire}(E')$ のとき $t_1' = \mathit{fire}(E')$ であり，

$$\begin{aligned} &\mathit{select}(\mathit{isData}(t_1'), t_1', E', t_3\theta) \\ =\ &\mathit{select}(\mathit{isData}(\mathit{fire}(E')), \mathit{fire}(E'), E', t_3\theta) \\ \to\ &\mathit{select}(\mathit{fire}(E'), \mathit{fire}(E'), E', t_3\theta) \\ \to\ &t_3\theta \\ \to^*\ &\mathit{fire}(E) \end{aligned}$$

$t_1\theta \to^* \mathit{fire}(E')$ なので，帰納法の仮定 (2) より $\mathit{ev}(t_1,\theta) = E'{\uparrow}$．したがって ev の定義より $\mathit{ev}(t,\theta) = \mathit{ev}(t_3,\theta)$．帰納法の仮定 (2) より $\mathit{ev}(t_3,\theta) = E{\uparrow}$．したがって $\mathit{ev}(t,\theta) = E{\uparrow}$．

- $\mathit{isData}(t_1') \to \mathit{fire}(E)$ かつ $E \ne E'$ のとき，$t_1' = \mathit{fire}(E)$ であり，

$$\begin{aligned} &\mathit{select}(\mathit{isData}(t_1'), t_1', E', t_3\theta) \\ =\ &\mathit{select}(\mathit{isData}(\mathit{fire}(E)), \mathit{fire}(E), E', t_3\theta) \\ \to\ &\mathit{select}(\mathit{fire}(E), \mathit{fire}(E), E', t_3\theta) \\ \to\ &\mathit{fire}(E) \end{aligned}$$

このとき $t_1\theta \to^* \mathit{fire}(E)$ なので，帰納法の仮定 (2) より $\mathit{ev}(t_1,\theta) = E{\uparrow}$．したがって ev の定義より $\mathit{ev}(t,\theta) = E{\uparrow}$．

- $t = \mathit{if}(t_1,t_2,t_3)$ のとき，$t\theta = \mathit{if}(t_1\theta, t_1\theta, t_1\theta)$ である．$t\theta \to^* \mathit{fire}(E)$ となるのは以下の 3 つの場合がある．
  - $t_1\theta \to^* \mathit{True}$ のとき，$t\theta \to^* \mathit{if}(\mathit{True}, t_2\theta, t_3\theta) \to t_2\theta \to^* \mathit{fire}(E)$ である．帰納法の仮定 (1) より $\mathit{ev}(t_1,\theta) = \mathit{True}{\downarrow}$ であり，帰納法の仮定 (2) より $\mathit{ev}(t_2,\theta) = E{\uparrow}$ であるから，$\mathit{ev}(t,\theta) = E{\uparrow}$．
  - $t_1\theta \to^* \mathit{False}$ のとき，$t\theta \to^* \mathit{if}(\mathit{False}, t_2\theta, t_3\theta) \to t_3\theta \to^* \mathit{fire}(E)$ である．帰納法の仮定 (1) より $\mathit{ev}(t_1,\theta) = \mathit{False}{\downarrow}$ であり，帰納法の仮定 (2) より $\mathit{ev}(t_3,\theta) = E{\uparrow}$ であるから，$\mathit{ev}(t,\theta) = E{\uparrow}$．
  - $t_1\theta \to^* \mathit{fire}(E)$ のとき，$t\theta \to^* \mathit{if}(\mathit{fire}(E), t_2\theta, t_3\theta) \to \mathit{fire}(E)$ である．帰納法の仮定 (2) より $\mathit{ev}(t_1,\theta) = E{\uparrow}$．したがって，$\mathit{ev}(t,\theta) = E{\uparrow}$．
- $t = f(t_1,\ldots,t_n)$，$f \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_\mathcal{D}$ のとき，$t\theta \to^* \mathit{fire}(E)$ となるのは以下の 2 つの場合がある．
  - ある i（$1 \le i \le n$）について $t_i\theta \to^* \mathit{fire}(E)$ であるとき，

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta, t_2\theta,\ldots,t_n\theta) \\ &\to^* f_2(v_1, t_2\theta,\ldots,t_n\theta) \\ &\to^* \cdots \\ &\to^* f_i(v_1,v_2,\ldots,v_{i-1},t_i\theta,t_{i+1}\theta,\ldots,t_n\theta) \\ &\to^* f_i(v_1,v_2,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta) \\ &\to \mathit{guard}(\mathit{isData}(\mathit{fire}(E)), f_{i+1}(v_1,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta)) \\ &\to \mathit{guard}(\mathit{fire}(E), f_{i+1}(v_1,v_2,\ldots,v_{i-1},\mathit{fire}(E),t_{i+1}\theta,\ldots,t_n\theta)) \\ &\to \mathit{fire}(E) \end{aligned}$$

となる．ここで，$\to$ は最内書き換えであるため $1 \le j < i$ であるす

べての j について $v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$．したがって，帰納法の仮定 (4) より，$evArgs([t_1,\ldots,t_n],\theta) = E\uparrow$ である．よって，$ev(t,\theta) = E\uparrow$ である．

– $f_1, f_2, \ldots, f_{n+1}$ を経由して書き換えられる場合，

$$\begin{aligned} t\theta &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta) \\ &\to^* f_1(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to guard(isData(v_1), f_2(v_1,t_2\theta,\ldots,t_n\theta)) \\ &\to guard(tt, f_2(v_1,t_2\theta,\ldots,t_n\theta)) \\ &\to f_2(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to^* \cdots \\ &\to^* f_{n+1}(v_1,\ldots,v_n) \end{aligned}$$

さらに $f(l_1,\ldots,l_n) \to r \in Fundef \cup BuiltinDef$ と σ が存在し，$\sigma = match[v_1,\ldots,v_n][l_1,\ldots,l_n]$ かつ $f_{n+1}(v_1,\ldots,v_n) \to r\sigma$ なので，

$$\begin{aligned} f_{n+1}(v_1,\ldots,v_n) &\to r\sigma \\ &\to^* fire(E) \end{aligned}$$

ここで，$\to$ は最内書き換えであるため $1 \le i \le n$ であるすべての i について $v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$．よって $f(t_1,\ldots,t_n)\theta \to^* f_{n+1}(v_1,\ldots,v_n)$ なので，帰納法の仮定 (3) より，$evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ である．$r\sigma \to^* fire(E)$ であるから帰納法の仮定 (2) より $ev(r,\sigma) = E\uparrow$ である．したがって $ev(t,\theta) = E\uparrow$ である．

(3)　$f(t_1,\ldots,t_n) \to^* f_{n+1}(v_1,\ldots,v_n)$ かつ $1 \le i \le n$ について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ とする．$f_{n+1}(v_1,\ldots,v_n)$ となるのは以下の書き換え系列のみである．

$$\begin{aligned} f(t_1,\ldots,t_n)\theta &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta) \\ &\to^* f_1(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to guard(isData(v_1), f_2(v_1,t_2\theta,\ldots,t_n\theta)) \\ &\to guard(tt, f_2(v_1,t_2\theta,\ldots,t_n\theta)) \\ &\to f_2(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to^* \cdots \\ &\to^* f_{n+1}(v_1,\ldots,v_n) \end{aligned}$$

このとき，$1 \le i \le n$ について $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ なので，帰納法の仮定 (1) より $ev(t_i,\theta) = v_i\downarrow$ である．したがって，$evArgs$ の定義より $evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ である．

(4)　ある i $(1 \le i \le n)$ が存在して $f(t_1,\ldots,t_n) \to^* f_i(v_1,\ldots,v_{i-1},fire(E),t_{i+1},\ldots,t_n)$ かつ $1 \le j < i$ であるすべての j について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ とする．このとき $1 \le n$ であり，書き換え系列は以下の場合のみである．

$$\begin{aligned} f(t_1,\ldots,t_n)\theta &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta) \\ &\to^* f_2(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to^* \cdots \\ &\to^* f_i(v_1,v_2,\ldots,v_{i-1},t_i\theta,t_{i+1}\theta,\ldots,t_n\theta) \\ &\to^* f_i(v_1,v_2,\ldots,v_{i-1},fire(E),t_{i+1}\theta,\ldots,t_n\theta) \end{aligned}$$

このとき，$1 \le j < i$ について $t_j\theta \to^* v_j \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ なので，帰納法の仮定 (1) より $ev(t_j,\theta) = v_j\downarrow$ である．また，$t_i\theta \to^* fire(E)$ なので，帰納法の仮定 (2) より $ev(t_i,\theta) = E\uparrow$ である．したがって，$evArgs$ の定義より $evArgs([t_1,\ldots,t_n],\theta) = E\uparrow$ である．

□

定理 4.2 の証明

証明　補題 A.1.5 と補題 A.1.6 より定理 4.2 は成り立つ．□

A.2　定理 5.1（健全性）の証明

以下の補題 A.2.1 と補題 A.2.2 は $\phi(\mathcal{P})$ だけでなく，一般の CS-TRS の最内書き換えについて成り立つ．

補題 A.2.1　$\langle\mathcal{R},\mu\rangle$ を CS-TRS とすると，$s \trianglerighteq_\mu t \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} u$ ならば，ある t' について $s \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} t' \trianglerighteq_\mu u$

証明　$s \trianglerighteq_\mu t$ より $t = s|_p, p \in \mathcal{P}os^\mu(s)$．$t \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} u$ より $s[t]_p \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} s[u]_p$．また，$s[u]_p \trianglerighteq_\mu u$．よって，$s \underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} s[u]_p \trianglerighteq_\mu u$　□

補題 A.2.2　$\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ が停止性を持つならば，$(\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} \cup \triangleright_\mu)$ は停止性を持つ．

証明　$(\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} \cup \triangleright_\mu)$ が停止性を持たないとする．このとき，$\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ が停止性を持ち，ある $t_1,\ldots,t_n$ に対して，$t_1(\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} \cup \triangleright_\mu)t_2(\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle} \cup \triangleright_\mu)t_3\cdots$ という無限系列が存在する．ここで，$\triangleright_\mu$ は明らかに停止性を持つので，$\underset{in}{\to}_{\langle\mathcal{R},\mu\rangle}$ も $\triangleright_\mu$ も一方が無限に続くことはない．よって，

$t_{n_1} \underset{in}{\rightarrow}^{+}_{\langle\mathcal{R},\mu\rangle} t_{n_2} \rhd_\mu t_{n_3} \underset{in}{\rightarrow}^{+}_{\langle\mathcal{R},\mu\rangle} \cdots$

($1 \leq n_1 < n_2 < n_3 \cdots$) という系列が存在し，補題 A.2.1 より，$\underset{in}{\rightarrow}_{\langle\mathcal{R},\mu\rangle}$ の無限系列が存在するので，$\underset{in}{\rightarrow}_{\langle\mathcal{R},\mu\rangle}$ の停止性に矛盾する．□

補題 A.2.3　$evArgs([t_1,\ldots,t_n],\theta)$ が未定義ならば，$ev(t_i,\theta)$ が未定義となる t_i が存在する．

証明　n に関する帰納法で証明する．$evArgs([t_1,\ldots,t_n],\theta)$ が未定義とする．

- $n=0$ のときは，$evArgs([],\theta)$ は未定義でないので考えなくてよい．
- $n>0$ のとき，
  - $ev(t_1,\theta)=v\downarrow$ のとき，$evArgs([t_2,\ldots,t_n],\theta)$ は未定義である．帰納法の仮定より $ev(t_j,\theta)$（$2\leq j\leq n$）が未定義となる t_j が存在する．
  - $ev(t_1,\theta)=E\uparrow$ のときは $evArgs([t_1,\ldots,t_n],\theta)$ は未定義でないので，考えなくてよい．
  - $ev(t_1,\theta)=\bot$ のときは $evArgs([t_1,\ldots,t_n],\theta)$ は未定義でないので，考えなくてよい．
  - $ev(t_1,\theta)$ が未定義のとき，t_1 は題意を満たす．

□

補題 A.2.4　$ev(t,\theta)$ が未定義ならば，(1) または (2) または (3) を満たす．

(1)　ある u と θ' が存在して，$t\theta \rightarrow^+ u\theta'$ かつ $ev(u,\theta')$ が未定義である．

(2)　ある u が存在して，$t \rhd_\mu u$ かつ $ev(u,\theta)$ が未定義である．

(3)　ある s と u と θ' が存在して $t\theta \rightarrow^+ s\theta'$ かつ $s \rhd_\mu u$ かつ $ev(u,\theta')$ が未定義である．

証明　$ev(t,\theta)$ が未定義であるとする．

- $t=x\in\mathcal{V}$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．
- $t=succ(t')$ のとき，$ev(t',\theta)$ は未定義である．このとき，$1\in\mu(succ)$ より $t\rhd_\mu t'$ であり，(2) を満たす．
- $t=c\in(\mathcal{F}_\mathcal{C}\backslash\{succ\})$ のとき $ev(t,\theta)$ は未定義でないので考えなくてよい．
- $t=raise(t')$ のとき，$t'=E\in Except$ ならば $ev(t,\theta)=E\uparrow$ となり $ev(t,\theta)$ は未定義でないので考えなくてよい．$t'\notin Except$ ならば $ev(t,\theta)=\bot$ となり $ev(t,\theta)$ は未定義でないので考えなくてよい．
- $t=handle(t_1,t_2,t_3)$ のとき
  - $t_2=E\in Except$ のとき
    - $ev(t_1,\theta)=v\downarrow$ ならば，$ev(t,\theta)$ は未定義でないので考えなくてよい．
    - $ev(t_1,\theta)=E'\uparrow$ かつ $E\neq E'$ ならば，$ev(t,\theta)=E'\uparrow$ となり $ev(t,\theta)$ は未定義でないので考えなくてよい．
    - $ev(t_1,\theta)=E\uparrow$ ならば，定理 4.2 から $t_1\theta\rightarrow^* fire(E)$ より，

$$\begin{aligned} t\theta &= handle(t_1\theta,E,t_3\theta)\\ &\rightarrow^* handle(fire(E),E,t_3\theta)\\ &\rightarrow select(isData(fire(E)),fire(E),E,t_3\theta)\\ &\rightarrow select(fire(E),fire(E),E,t_3\theta)\\ &\rightarrow t_3\theta \end{aligned}$$

    となり，$t\theta\rightarrow^+ t_3\theta$．$ev(t,\theta)$ が未定義となるためには $ev(t_3,\theta)$ が未定義でなければならない．よって (1) を満たす．
    - $ev(t_1,\theta)=\bot$ ならば，$ev(t,\theta)=\bot$ となり，$ev(t,\theta)$ は未定義でないので考えなくてよい．
    - $ev(t_1,\theta)$ が未定義ならば，$1\in\mu(handle)$ より $t\rhd_\mu t_1$ であり，(2) を満たす．
  - $t_2\notin Except$ ならば $ev(t,\theta)=\bot$ となり $ev(t,\theta)$ は未定義でないので考えなくてよい．
- $t=if(t_1,t_2,t_3)$ のとき，
  - $ev(t_1,\theta)=True\downarrow$ のとき，定理 4.2 から $t_1\theta\rightarrow^* True$ より，
    $t\theta=if(t_1\theta,t_2\theta,t_3\theta)\rightarrow^* if(True,t_2\theta,t_3\theta)\rightarrow t_2\theta$
    となり，$t\theta\rightarrow^+ t_2\theta$．$ev(t,\theta)$ が未定義となるためには $ev(t_2,\theta)$ が未定義でなければならない．よって (1) を満たす．
  - $ev(t_1,\theta)=False\downarrow$ のとき，定理 4.2 から $t_1\theta\rightarrow^* False$ より，
    $t\theta=if(t_1\theta,t_2\theta,t_3\theta)\rightarrow^* if(False,t_2\theta,t_3\theta)\rightarrow t_3\theta$
    となり，$t\theta\rightarrow^+ t_3\theta$．$ev(t,\theta)$ が未定義となるためには $ev(t_3,\theta)$ が未定義でなければならない．よって (1) を満たす．
  - $ev(t_1,\theta)=v\downarrow$ かつ $v\neq True$ かつ $v\neq False$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．
  - $ev(t_1,\theta)=E\uparrow$ ならば $ev(t,\theta)$ は未定義でないので考えなくてよい．
  - $ev(t_1,\theta)=\bot$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．
  - $ev(t_1,\theta)$ が未定義ならば，$1\in\mu(if)$ より $t\rhd_\mu t_1$ であり，(2) を満たす．
- $t=f(t_1,\ldots,t_n)$ かつ $f\in\mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_\mathcal{D}$ のとき，

– $evArgs([t_1,\ldots,t_n],\theta) = [v_1,\ldots,v_n]\downarrow$ かつ $f(l_1,\ldots,l_n) \to r \in Fundef \cup BuiltinDef$ かつ $\sigma = match[v_1,\ldots,v_n][l_1,\ldots,l_n]$ ならば，定理 4.2 から各 i に対して $t_i\theta \to^* v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ より，

$$\begin{aligned} t\theta = f(t_1\theta,\ldots,t_n\theta) &\to^* f(v_1,\ldots,v_n) \\ &= f(l_1\sigma,\ldots,l_n\sigma) \\ &= f(l_1,\ldots,l_n)\sigma \\ &\to r\sigma \end{aligned}$$

となり，$t\theta \to^+ r\sigma$．$ev(t,\theta)$ が未定義となるためには $ev(r,\sigma)$ が未定義でなければならない．よって (1) を満たす．

– $evArgs([t_1,\ldots,t_n],\theta) = E\uparrow$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．

– $evArgs([t_1,\ldots,t_n],\theta) = \bot$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．

– $evArgs([t_1,\ldots,t_n],\theta)$ が未定義ならば，補題 A.2.3 より，$ev(t_i,\theta)$ が未定義となる t_i が存在する．このとき $1 \le j < i$ であるすべての j について $t_j\theta \to v_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ であり $t\theta$ は以下のように書き換えられる．

$$\begin{aligned} t\theta &= f(t_1,\ldots,t_n)\theta \\ &= f(t_1\theta,\ldots,t_n\theta) \\ &\to f_1(t_1\theta,t_2\theta,\ldots,t_n\theta) \\ &\to^* f_2(v_1,t_2\theta,\ldots,t_n\theta) \\ &\to^* \cdots \\ &\to^* f_i(v_1,v_2,\ldots,v_{i-1},t_i\theta,t_{i+1}\theta,\ldots,t_n\theta) \\ &= f_i(v_1,v_2,\ldots,v_{i-1},t_i,t_{i+1},\ldots,t_n)\theta \end{aligned}$$

$i \in \mu(f_i)$ より $f_i(v_1,v_2,\ldots,v_{i-1},t_i,t_{i+1},\ldots,t_n) \rhd_\mu t_i$ であり，(3) を満たす．

• $t \in Except$ のとき，$ev(t,\theta)$ は未定義でないので考えなくてよい．

以上より，$ev(t,\theta)$ が未定義ならば，(1) または (2) または (3) を満たす． □

定理 5.1 の証明

以下に停止性に関するの健全性の定理 5.1，すなわち $\phi(\mathcal{P})$ が最内停止性を持つならば $\mathcal{P}$ が停止性を持つことの証明を与える．

証明　$\mathcal{P}$ が停止性を持たないなら $\phi(\mathcal{P})$ も最内停止性を持たないことを示す．ある t と θ について $ev(t,\theta)$ が未定義のとき，$t\theta$ から始まる $\to$ の無限書き換え系列が存在することを示す．$ev(t,\theta)$ が未定義ならば，補題 A.2.4 より $t\theta(\to \cup \rhd_\mu)t_1\theta_1(\to \cup \rhd_\mu)t_2\theta_2\cdots$ という無限系列が存在する．このとき，補題 A.2.1，補題 A.2.2 より $t\theta$ から始まる $\to$ の無限系列を作ることができる． □

A.3　定理 5.2 （完全性）の証明

補題 A.3.1　$t \to^* s$ であるとき $t \in ISN$ ならばかつそのときに限り $s \in ISN$

証明　**(⇐)**　補題 A.1.2 より明らか．

(⇒)　定義より明らか． □

補題 A.3.2　$t \to^* s$ かつ $s \in NF$ ならば $t \in ISN$

証明　補題 A.3.1 より明らか． □

補題 A.3.3　$\mathcal{P}$ が停止性を持つとする．$\phi(\mathcal{P})$ において，$root(t) = g \in \mathcal{F}$ を満たす項 $t = g(t_1,\ldots,t_n) \in \mathcal{T}(\mathcal{F}')$ について $t \notin ISN$ ならば $t_i \notin ISN$ を満たす t_i が存在する．すなわち以下が成り立つ．

(1)　$succ(t_1) \notin ISN$ ならば $t_1 \notin ISN$.

(2)　$raise(t_1) \notin ISN$ ならば $t_1 \notin ISN$.

(3)　$handle(t_1,t_2,t_3) \notin ISN$ ならば，$t_1 \notin ISN$ と $t_2 \notin ISN$ と $t_3 \notin ISN$ のいずれかが成り立つ．

(4)　$if(t_1,t_2,t_3) \notin ISN$ ならば，$t_1 \notin ISN$ と $t_2 \notin ISN$ と $t_3 \notin ISN$ のいずれかが成り立つ．

(5)　$g \in \mathcal{F}_{\mathcal{B}'} \cup \mathcal{F}_\mathcal{D}$ について，$g(t_1,\ldots,t_n) \notin ISN$ ならば $t_i \notin ISN$ となる t_i が存在する．

証明　対偶，すなわち $t_i \in ISN$ $(1 \le i \le n)$ ならば $t = g(t_1,\ldots,t_n) \in ISN$ を示す．

(1)　$succ(t_1)$ において $t_1 \in ISN$ とする．このとき，$t_1\Downarrow = u_1$ となる u_1 が存在し，$succ(t_1) \to succ(u_1)$ となる．

- $u_1 = fire(E)$ かつ $E \in Except$ のとき，$succ(fire(E)) \to fire(E) \in NF$.
- それ以外のとき，$succ(u_1) \in NF$.

いずれの場合も補題 A.3.2 より $succ(t_1) \in ISN$ である．

(2)　$t = raise(t_1)$ のとき，

- $t_1 \in Except$ のとき，$raise(t_1) \to fire(t_1) \in NF$.
- $t_1 \notin Except$ のとき $raise(t_1) \in NF$.

いずれの場合も補題 A.3.2 より $raise(t_1) \in ISN$ である．

(3)　$handle(t_1,t_2,t_3)$ において，$t_1 \in ISN$ かつ $t_2 \in ISN$ かつ $t_3 \in ISN$ とする．このとき，$t_1\Downarrow = u_1$，$t_2\Downarrow = u_2$，$t_3\Downarrow = u_3$ となる u_1，u_2，u_3 が存在し，$handle(t_1,t_2,t_3) \to^* handle(u_1,t_2,t_3)$ となる．

- $t_2 = E \in Except$ のとき $handle(u_1,E,t_3) \to select(isData(u_1),u_1,E,t_3)$.
  – $u_1 \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ のとき

$select(isData(u_1), u_1, E, t_3) \to^* select(tt, u_1, E, t_3) \to u_1 \in NF$

– $u_1 = fire(s)$ のとき

$select(isData(fire(s)), fire(s), E, t_3) \to select(fire(s), fire(s), E, t_3)$.

＊ $s = E$ のときは $select(fire(E), fire(E), E, t_3) \to t_3 \to^* u_3 \in NF$

＊ $s \neq E$ かつ $s \in Except$ のときは

$select(fire(s), fire(s), t_2, t_3) \to fire(s) \in NF$

＊ $s \notin Except$ のときは $select(fire(s), fire(s), t_2, t_3) \in NF$

– $u_1 \notin \mathcal{T}(\mathcal{F}_\mathcal{C})$ かつ $u_1 \neq fire(s)$ かつ $u_1 = succ^k(u_1')$（$k > 1$）かつ $root(u_1') \neq succ$ のとき，

$select(isData(u_1), u_1, t_2, t_3) \to^* select(isData(u_1'), u_1, t_2, t_3) \in NF$.

– u_1 がその他の項のとき $select(isData(u_1), u_1, t_2, t_3) \in NF$.

- $t_2 \notin Except$ のとき $handle(u_1, t_2, t_3) \in NF$.

よっていずれの場合も補題 A.3.2 より $handle(t_1, t_2, t_3) \in ISN$ である．

（4） $if(t_1, t_2, t_3)$ において，$t_1 \in ISN$ かつ $t_2 \in ISN$ かつ $t_3 \in ISN$ とする．このとき，$t_1\Downarrow = u_1$，$t_2\Downarrow = u_2$，$t_3\Downarrow = u_3$ となる u_1，u_2，u_3 が存在し，$if(t_1, t_2, t_3) \to^* if(u_1, t_2, t_3)$.

- $u_1 = True$ のとき $if(True, t_2, t_3) \to t_2 \to^* u_2 \in NF$
- $u_1 = False$ のとき $if(False, t_2, t_3) \to t_3 \to^* u_3 \in NF$
- $u_1 = fire(s)$ のとき $if(u_1, t_2, t_3) \to fire(s) \in NF$
- u_1 がその他の項のとき $if(u_1, t_2, t_3) \in NF$.

よっていずれの場合も補題 A.3.2 より $if(t_1, t_2, t_3) \in ISN$ である．

（5） $g(t_1, \ldots, t_n)$ において，各 i（$1 \leq i \leq n$）について $t_i \in ISN$ とする．このとき各 $t_i\Downarrow$ が定まり，それぞれ u_i とおくことにする．

- すべての i（$1 \leq i \leq n$）に対して，$isData(t_i\Downarrow) \to^* tt$ のとき $t_i\Downarrow = u_i \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ である．このとき $g(t_1, \ldots, t_n) \to^* g_{n+1}(u_1, \ldots, u_n)$ となる．もし $t \notin ISN$ ならば補題 A.3.1 より $g_{n+1}(u_1, \ldots, u_n) \notin ISN$ である．したがって，$g_{n+1}(u_1, \ldots, u_n)$ が左辺とマッチする書き換え規則 $l \to r$ が存在する．$l\theta = g_{n+1}(u_1, \ldots, u_n)$ とすると，すべての $x \in \mathcal{V}ar(l)$ に対して $u_i \trianglerighteq x\theta$ であり，$x\theta \in \mathcal{T}(\mathcal{F}_\mathcal{C}) \subseteq \mathcal{T}(\mathcal{F})$ である．このとき，$r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ であるから $r\theta \in \mathcal{T}(\mathcal{F})$ である．$\mathcal{P}$ は停止性を持つため $ev(r, \theta)$ の値が定まり，補題 A.1.5 より $r\theta \in ISN$ である．したがって，$g(t_1, \ldots, t_n) \in ISN$ である．
- ある i（$1 \leq i \leq n$）に対して $isData(u_i) \to^* fire(s)$ かつ $1 \leq j < i$ であるすべての j について $isData(u_j) \to^* tt$ のとき $g(t_1, \ldots, t_n) \to^* fire(s) \in NF$
- ある i（$1 \leq i \leq n$）に対して，$isData(u_i)\Downarrow = u_i'$ かつ $u_i' \neq tt$ かつ $u_i' \neq fire(s)$ であり，$1 \leq j < i$ であるすべての j について $isData(u_j) \to^* tt$ のとき

$$\begin{aligned} g(t_1, \ldots, t_n) &\to^* g_i(u_1, \ldots, u_{i-1}, u_i, t_{i+1}, \ldots, t_n) \\ &\to guard(isData(u_i), g_{i+1}(u_1, \ldots, u_i, t_{i+1}, \ldots, t_n)) \\ &\to^* guard(u_i', g_{i+1}(u_1, \ldots, u_i, t_{i+1}, \ldots, t_n)) \in NF \end{aligned}$$

よっていずれの場合も補題 A.3.2 より $g(t_1, \ldots, t_n) \in ISN$ である．

□

定理 5.2 の証明

以下に停止性に関する完全性の定理 5.2，すなわち $\mathcal{P}$ が停止性を持つならば $\phi(\mathcal{P})$ が最内停止性を持つことの証明を与える．定義 3.1 で与えたように $\mathcal{F}$ はプログラム $\mathcal{P}$ に現れる関数記号の集合であり，定義 4.1 で与えたように $\mathcal{F}'$ は CS-TRS $\phi(\mathcal{P})$ に現れる関数記号の集合である．また，$\phi(\mathcal{P})$ による文脈依存最内書き換えは一意に決まる．項の集合 T_m^∞ を $\{t \mid t \in \mathcal{T}(\mathcal{F}') \wedge t \notin ISN \wedge \forall u \lhd t, u \in ISN\}$ と定義する．このとき，$\phi(\mathcal{P})$ が最内停止性を持つとき，かつそのときに限り $T_m^\infty = \emptyset$ である．

証明　$\mathcal{P}$ が停止し，かつ $T_m^\infty \neq \emptyset$ を仮定し，矛盾を導く．$t \in T_m^\infty$ とする．このとき $t \notin ISN$ であり，また t のどの真部分項も ISN に属する．

- $root(t) = g \in \mathcal{F}$ のとき，$t \notin ISN$ なので，補題 A.3.3 より g の引数に $u \notin ISN$ が存在する．一方，$t \rhd u$ より $u \in ISN$ であり矛盾する．
- $root(t) = tt$ のとき $t \in NF \subseteq ISN$ であり，考えなくてよい．
- $root(t) = fire$ のとき $t \in NF \subseteq ISN$ であり，考えなくてよい．
- $root(t) = guard$ のとき，$t = guard(u_1, u_2)$ とすると，$u_1, u_2 \in ISN$ である．
  – $u_1\Downarrow = tt$ のとき，$t \to^* u_2 \in ISN$ である．一方，補題 A.3.1 より $u_2 \notin ISN$ であり矛盾．
  – $u_1\Downarrow = fire(s)$ のとき，$t \to^* fire(s) \in ISN$ である．一方，補題 A.3.1 より $fire(s) \notin ISN$ であり矛盾．
  – $u_1\Downarrow \neq tt \wedge u_1\Downarrow \neq fire(s)$ のとき，$t \to^* guard(u_1\Downarrow, u_2) \in NF \subseteq ISN$ であり，矛盾．
- $root(t) = select$ のとき $t = select(u_1, u_2, u_3, u_4)$ とすると，$u_1, u_2, u_3, u_4 \in ISN$ である．
  – $u_1\Downarrow = tt$ のとき，$t \to^* u_2 \in ISN$ となり矛盾．

– $u_1\Downarrow = \mathit{fire}(E)$ かつ $E \in \mathit{Except}$ かつ $u_3 = E' \in \mathit{Except}$ のとき
  * $E = E'$ のとき $t \to^* u_4 \in \mathit{ISN}$ となり矛盾．
  * $E \neq E'$ のとき $t \to^* u_1 \in \mathit{ISN}$ となり矛盾．
– それ以外のとき $t \to^* \mathit{select}(u_1\Downarrow, u_2, u_3, u_4) \in \mathit{NF}$ となり矛盾．

- $\mathit{root}(t) = \mathit{isData}$ のとき $t = \mathit{isData}(u)$ とすると，$\mu(\mathit{isData}) = \mu(\mathit{fire}) = \emptyset$ より
  – $u \in \mathcal{T}(\mathcal{F}_\mathcal{C})$ のとき，$t \to^* tt \in \mathit{NF}$
  – $u = \mathit{fire}(s)$ のとき，$t \to \mathit{fire}(s) \in \mathit{NF}$
  – $u \notin \mathcal{T}(\mathcal{F}_\mathcal{C})$ かつ $u = \mathit{succ}^k(u')$（$k > 1$）かつ $\mathit{root}(u') \neq \mathit{succ}$ のとき，$t \to^* \mathit{isData}(u') \in \mathit{NF}$
  – それ以外のとき，$t \in \mathit{NF}$

  いずれの場合も補題 A.3.2 より $t \in \mathit{ISN}$ であり，$t \notin \mathit{ISN}$ に矛盾する．
- $\mathit{root}(t) = f_i \in \mathcal{F}_\mathcal{E}$，$\mathit{arity}(f_i) = n$ のとき，$t = f_i(t_1, \ldots, t_n)$ とする．各 $t_i \in \mathit{ISN}$ であり，$\mu(f_j) = \{j\}$（$1 \leq j \leq n$）より t は次のような書き換え系列によって書き換えられる．

$$\begin{aligned} t &\to^* f_{i+1}(t_1, \ldots, t_{i-1}, t_i\Downarrow, t_{i+1}, \ldots, t_n) \\ &\to^* \cdots \\ &\to^* f_{n+1}(t_1, \ldots, t_{i-1}, t_i\Downarrow, t_{i+1}\Downarrow, \ldots, t_n\Downarrow) \end{aligned}$$

  このとき $t_j \in \mathit{ISN}$（$1 \leq j \leq i-1$）であり，$t_k\Downarrow \in \mathit{NF}$（$i \leq k \leq n$）である．$t_j = u_j$，$t_k\Downarrow = u_k$ とおくと，$u_l \in \mathit{ISN}$（$1 \leq l \leq n$）である．$\mu(f_{n+1}) = \emptyset$ より，さらに書き換えが続くためには $f_{n+1}(l_1, \ldots, l_n) \to r \in \mathit{Fundef}'$ となる規則，$\sigma = \mathit{match}[u_1, \ldots, u_n][l_1, \ldots, l_n]$ となる代入 σ が存在しなければならない．このとき $f_{n+1}(u_1, \ldots, u_n) \to r\sigma$ であり，$r\sigma \notin \mathit{ISN}$ なら $r' \trianglelefteq r$ かつ $r'\sigma \in T_m^\infty$ となる r' が存在する．ここで，任意の $x \in \mathcal{V}ar(r') \subseteq \mathcal{V}ar(r)$ について，ある i（$1 \leq i \leq n$）が存在して $x\sigma \trianglelefteq u_i$ である．よって，$x\sigma \in \mathit{ISN}$．$r' \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ より，補題 A.3.3 を繰り返し使うことで $r'\sigma \in \mathit{ISN}$ を示すことができ，$r\sigma \in \mathit{ISN}$ である．補題 A.3.1 より $t \in \mathit{ISN}$ となり $t \notin \mathit{ISN}$ に矛盾する．

□

(平成 22 年 9 月 28 日受付)
(平成 22 年 12 月 21 日採録)

濱口　毅

1995 年名古屋大学大学院工学研究科情報工学専攻博士課程満了．1995 年同大学工学部助手．2007 年名古屋大学大学院情報科学研究科助教，現在に至る．電子情報通信学会，ソフトウェア科学会各会員．

酒井　正彦

1989 年名古屋大学大学院博士課程満了．同年同大学工学部助手．1993 年北陸先端科学技術大学院大学助教授．1997 年名古屋大学大学院工学研究科助教授．2002 年同教授．2003 年同大学院情報科学研究科教授，現在に至る．この間，1996 年 3〜8 月ニューヨーク州立大学スートーニーブルック校客員研究教授．項書き換え系等のソフトウェア基礎理論に関する研究に従事．工学博士．平成 3 年度電子情報通信学会論文賞受賞．日本ソフトウェア科学会会員．

馬場　正貴

2010 年名古屋大学大学院情報科学研究科計算機数理科学専攻博士課程前期課程修了．同年 KDDI 株式会社入社．

阿草　清滋（正会員）

1970 年京都大学工学部電気第二学科卒業，1972 年同大学大学院工学研究科電気工学第二専攻修士課程修了，同博士課程進学．1974 年京都大学情報工学科助手．同講師，助教授を経て 1989 年より名古屋大学教授．工学博士．ソフトウェア開発方法論，知的開発環境，仕様化技法，再利用技法，マンマシンインタフェース等の研究に従事．