一般社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SS2013-23, KBSE2013-23 (2013-07)

例外処理を含む関数型プログラム停止性証明のための条件付き依存対法

濱口 毅† 酒井 正彦†

† 名古屋大学大学院情報科学研究科
〒 464-8601 愛知県名古屋市千種区不老町
E-mail: †{hamaguti,sakai}@is.nagoya-u.ac.jp

あらまし 先に提案した文脈依存項書換え系（CS-TRS）への変換による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・非停止性証明法では，変換で得られる CS-TRS の停止性・非停止性証明に汎用の停止性証明ツールを利用すると非常に短いプログラムしか証明に成功しない．そこで，本論文では例外処理を持つ関数型プログラムから変換された CS-TRS の停止性証明のための新しい手法を提案する．まず，項書換え系（TRS）の停止性証明に用いられる依存対を拡張し，文脈を条件として記述する条件付き依存対を定義する．次に，条件付き依存対から構成される条件付き依存対鎖の存在と CS-TRS の最内停止性が一致することを証明する．さらに，依存グラフを用いた既存の手法を拡張し，条件付き依存対グラフによる CS-TRS の停止性判定手法を提案する．本手法によりこれまで証明ができなかった多くのプログラムの停止性・非停止性が証明可能となる．

キーワード 関数型プログラム，例外処理，項書換え系，停止性

Conditional Dependency Pair Method for Proving Termination of Functional Programs with Exception Handling

Takeshi HAMAGUCHI† and Masahiko SAKAI†

† Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601 Japan
E-mail: †{hamaguti,sakai}@is.nagoya-u.ac.jp

Abstract We have recently proposed a method for proving termination/non-termination properties of eager-evaluation-based functional programs with exception handling. The method transforms them into Context-Sensitive Term Rewriting Systems (CS-TRSs) in preserving the properties. However we encounter a problem that the existing termination provers for CS-TRSs fail even if a very short program is given. In this paper, we present a dependency method specialized for CS-TRSs transformed from functional programs with exception handling. We introduce conditions that represent context information into the dependency pairs, and define conditional dependency chains. We prove that the target CS-TRS is inner-most terminating if and only if there exists no infinite conditional dependency chain. Moreover, we augment graph notion into the framework of the dependency pair problems, and propose some new processors. The new method works effectively for CS-TRSs produced by the transformation.

Key words functional program, exception handling, term rewriting system, and termination

1. まえがき

すでに我々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した [1]．対象プログラミング言語は *ML/ex* とし，*ML/ex* を文脈依存項書換え系（CS-TRS）に変換して停止性の判定を行う．この変換は停止性に対して健全かつ完全であるため，CS-TRS の最内書換えによる停止性・非停止性を証明することで，プログラムの停止性・非停止性が証明できる．CS-TRS の停止性・非停止性には既存の停止性証明ツール [2], [3] を用いていたが，文脈依存と最内の組み合わせにおいては停止性証明の能力が弱く，証明できたプログラムは非常に短い限られたものであった．

そこで，本論文では例外処理を持つ関数型プログラムから変換された CS-TRS に有効な停止性証明法を提案する．項書換え系（TRS）の停止性証明の手法として依存対法 [4]，依存グラフ [5] が広く知られている．項の到達性を条件として記述可能

な条件付き依存対 (*CDP*,Conditional Dependcy Pair) の定義を与える．次に [1] で得られる CS-TRS の *CDP* 集合への変換法を与え，*CDP* から構成される無限の条件付き依存対鎖の非存在と CS-TRS の最内停止性が一致することを示す．この効果的な証明のため，依存対を用いた項書換え系の停止性の判定手法として提案された DP プロセッサを用いた手法 [4] の適用を図る．*CDP* 列が鎖となるためにはそれらのすべての条件を成立させる代入の存在が必要となるため，制約付き項書換え系の停止性判定のための GDP 問題とそのプロセッサ [6] のように，DP プロセッサの入力に依存グラフの情報を加えることが効果的であることが分かった．そこで，GDP 問題のように依存グラフの情報を追加した *CDP* 問題を定義し，関数型プログラムから生成された *CDP* 問題例に効果的なプロセッサを提案し，その停止性証明例を示す．

2. 準　　備

本節では本論文で用いる記法を説明する．概念の詳細は文献 [7] などを参照されたい．

固定したアリティを持つ**関数記号**からなる有限集合を $\mathcal{F}$ とする．$\mathcal{F}$ は**構成子**の集合 C と**被定義記号**の集合 D に分割されるものとする．関数記号に対して，その引数の数を表す関数 $arity$ と**文脈**を表す関数 μ があるとする．ここで $\mu(f)\subseteq\{1,\ldots,arity(f)\}$ を満たす．$\mathcal{F}$ と可算無限個の**変数**の集合 $\mathcal{V}$ から生成されるすべての $\mathcal{F}$ **項** (あるいは項) の集合を $\mathcal{T}(\mathcal{F},\mathcal{V})$ と，$\mathcal{T}(\mathcal{F},\emptyset)$ を $\mathcal{T}(\mathcal{F})$ と，項 $t_1,\ldots,t_n$ のリストを $[t_1,\ldots,t_n]$ と書く．項 t に現れるすべての変数の集合を $\mathcal{V}ar(t)$ で表す．項を木と捉えることにより，出現する記号の**位置**を通常どおり自然数列で表す．例えば $t=f(g(a),b)$ において，a は位置 11 に出現するという．項 t に対する μ **置換位置** $\mathcal{P}os^{\mu}(t)$ は (1) $x\in\mathcal{V}$ について，$\mathcal{P}os^{\mu}(x)=\{\varepsilon\}$，(2) $\mathcal{P}os^{\mu}(f(t_1,\ldots,t_n))=\{\varepsilon\}\cup\{ip\mid i\in\mu(f),p\in\mathcal{P}os^{\mu}(t_i)\}$ として再帰的に定義される．根の位置 ε に出現する記号 f を $root(t)$ で表す．項 t の μ 置換位置 p における部分項を μ **部分項**と呼び $t|_p^{\mu}$ で表す．それ自身以外の部分項を真部分項という．t の p における部分項を項 s で置き換えて得られる項を $t[s]_p^{\mu}$ で表す．

代入 θ の項 t に適用して得られる項を $t\theta$ で表す．$\theta:\mathcal{V}\to\mathcal{T}(\mathcal{F},\mathcal{V})$ を θ の定義域と呼び，定義域が空の代入を $\emptyset$ と書く．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．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．

$\mathcal{F}=C\uplus D$ 上の書き換え規則 (l,r) は $root(l)\in D$，$l,r\in\mathcal{T}(\mathcal{F},\mathcal{V})$，$\mathcal{V}ar(l)\supseteq\mathcal{V}ar(r)$ を満たす項の対 l,r であり，$l\to r$ で表す．

$\mathcal{F}$ 上の**書換え規則の集合** $\mathcal{R}$ を $\mathcal{F}$ 上の**文脈依存項書換え系** (*CS-TRS*) という．ある $l\to r\in\mathcal{R}$ について，$t|_p^{\mu}=l\theta$ を満たすとき，$l\theta$ を t の**リデックス**という．さらに $l\theta$ が $l\theta$ のみをリデックスとして持つとき，$l\theta$ は t の**最内リデックス**という．項 s の (最内) リデックスが $l\theta$ であるとき，$s=s[l\theta]_p^{\mu}$ は $t=s[r\theta]_p^{\mu}$ に (**最内**) **書換え**されるという．s が t に書換えられる，あるいは最内書換えされることを，それぞれ，$s\to_{\mathcal{R}}t$，$s\underset{in}{\to}_{\mathcal{R}}t$ と書く．また，リデックスを持たない項を**正規形**という．また，$s\underset{in}{\to}_{\mathcal{R}}t$ かつ t が正規形であるとき，t は s の**最内正規形**という．$t\in\mathcal{T}(\mathcal{F},\mathcal{V})$ から始まる無限の $\to_{\mathcal{R}}$ ($\underset{in}{\to}_{\mathcal{R}}$) による書換え系列が存在しないとき，$t$ は $\mathcal{R}$ のもとで**停止性** (**最内停止性**) を持つといい，どの項も $\mathcal{R}$ のもとで (最内) 停止性を持つとき，$\mathcal{R}$ は (最内) 停止性を持つという．

関係 $\rightharpoonup$ の推移閉包，反射推移閉包を，それぞれ，$\rightharpoonup^+$ と $\rightharpoonup^*$ で表す．文脈 μ がどの関数記号 $f\in\mathcal{F}$ についても $|\mu(f)|=arity(f)$ を満たすとき，$\mathcal{R}$ は (文脈に依存しない) 項書換え系である．

3. 条件付き依存対と条件付き依存対問題

次節で述べる関数型プログラムから変換で得られた CS-TRS の停止性を効果的に判定するために，本節では依存対に条件を導入し，[6] の GDP プロセッサ手法と同様な依存グラフを導入した依存対問題を定義する．

通常の項書換え系に対する依存対法と同様に，関数記号 $f\in D$ に対して**組記号**と呼ばれる新しい記号 $f^{\sharp}$ を導入する．すなわち，$D^{\sharp}=\{f^{\sharp}\mid f\in D\}$ とし，$\mathcal{F}^{\sharp}=\mathcal{C}\uplus\mathcal{D}\uplus\mathcal{D}^{\sharp}$ とする．ここで，$arity(f^{\sharp})=arity(f)$，$\mu(f^{\sharp})=\mu(f)$ とする．項 $t=f(t_1,\ldots,t_n)\in\mathcal{T}(\mathcal{F},\mathcal{V})$ に対して根記号 f を $f^{\sharp}$ で置き換えて得られる項を $t^{\sharp}$ で表す．文脈依存性を扱うために依存対に関係論理の式を導入する．

[定義 3.1]　**項上の関係論理**の**論理式**は，項の対 $s\twoheadrightarrow t$ を**原始式**とし，原始式と論理積で構成される．その論理式 F の値は，項上の関係 $\rightharpoonup$ と代入から以下のように定められる，

$$\langle\rightharpoonup,\sigma\rangle\models s\twoheadrightarrow t\iff s\sigma\rightharpoonup t\sigma$$
$$\langle\rightharpoonup,\sigma\rangle\models F\wedge F'\iff\langle\rightharpoonup,\sigma\rangle\models F\text{ and }\langle\rightharpoonup,\sigma\rangle\models F'$$

与えられた $\rightharpoonup$ のもとで，$\langle\rightharpoonup,\sigma\rangle\models F$ が真となる代入 σ が存在するとき，F は $\rightharpoonup$ **充足可能**であるという．そうでないとき，$\rightharpoonup$ **充足不能**であるという．

[定義 3.2] (条件付き依存対)　$\mathcal{F}^{\sharp}$ 項 $s^{\sharp},t^{\sharp}$ と $\mathcal{F}$ 項上の関係論理式 c の対 $(\langle s^{\sharp},t^{\sharp}\rangle,c)$ を**条件付き依存対** (*CDP*) と呼ぶ．ここで c を条件部と呼ぶ．

直感的には $CDP(\langle s^{\sharp},t^{\sharp}\rangle,c)$ は代入 σ に対して $c\sigma$ が成立するとき項 $s\sigma$ が 1 回以上の書換えで項 $t\sigma$ に書き換えることができることを表す．

[定義 3.3] (条件付き依存対鎖)　$\mathcal{R}$ を $\mathcal{F}$ 上の CS-TRS とする．*CDP* の列 $(\langle s_1^{\sharp},t_1^{\sharp}\rangle,c_1)(\langle s_2^{\sharp},t_2^{\sharp}\rangle,c_2)\cdots$ は，以下の条件をすべて満たす代入 $\sigma_1,\sigma_2,\ldots$ が存在するとき**条件付き依存対鎖** (*CDP* 鎖) という．

(1)　$t_i^{\sharp}\sigma_i\to_{\mathcal{R}}^{*}s_{i+1}^{\sharp}\sigma_{i+1}$

(2)　$\langle\to_{\mathcal{R}}^{*},\sigma_i\rangle\models c_i$

(3)　$s_i^{\sharp}\sigma_i$ は $\mathcal{R}$ のもとで停止性を持つ．

また，(1),(2) 中の $\to_{\mathcal{R}}^{*}$ を $\underset{in}{\to}_{\mathcal{R}}^{*}$ で置き換えた条件を満たし，(3)' $s_i^{\sharp}\sigma_i$ が正規形であるような代入 $\sigma_1,\sigma_2,\ldots$ が存在するとき，これを**最内条件付き依存対鎖** (最内 *CDP* 鎖) と呼ぶ．

[定義 3.4] (条件付き依存対グラフ)　$\mathcal{R}$ を $\mathcal{F}$ 上の CS-TRS,

V を CDP の集合とする．$\mathcal{G} = (V, E)$ を**条件付き依存対グラフ**（CDP グラフ）と呼ぶ．また，グラフの経路上の頂点列が（最内）CDP 鎖であるとき，これを $\mathcal{G}$ 上の（最内）CDP 鎖と呼ぶ．

[定義 3.5]（(最内) 条件付き依存対問題）（最内）条件付き依存対問題（CDP 問題）とは CDP グラフ $\mathcal{G}$ と CS-TRS $\mathcal{R}$ の対を入力とし，$\mathcal{G}$ 上の無限の（最内）CDP 鎖が存在しないかを判定する問題である．$(\mathcal{G}, \mathcal{R})$ を（最内）CDP 問題の入力とするとき，$\mathcal{G}$ 上の無限の（最内）CDP 鎖が存在するとき，かつそのときに限り $(\mathcal{G}, \mathcal{R})$ は**有限**であるという．

以下では CDP 問題の入力 $(\mathcal{G}, \mathcal{R})$ を単に CDP 問題と呼ぶ．無限の（最内）CDP 鎖の存在性は（最内）CDP 問題に変換できる．

[補題 3.6] $\mathcal{R}$ を CS-TRS，V を CDP とする．このとき，(最内) CDP 問題 $(V, \{(e, e') \mid e, e' \in V\}), \mathcal{R})$ が有限のとき，かつそのときに限り，V の要素の無限列で，$\mathcal{R}$ に対する（最内）CDP 鎖となるものは存在しない．

この問題の変換を行う（最内）条件付き依存対プロセッサを定義する．

[定義 3.7]（(最内) 条件付き依存対プロセッサ）**(最内) 条件付き依存対プロセッサ** (CDP プロセッサ) $Proc$ は（最内）CDP 問題を入力とし，(最内) CDP 問題の集合または $\bot$ を返す関数である．

[定義 3.8]（(最内) 条件付き依存対プロセッサの健全性）$Proc(\mathcal{G}, \mathcal{R}) \neq \bot$ かつどの $(\mathcal{G}_i, \mathcal{R}_i) \in Proc(\mathcal{G}, \mathcal{R})$ についても $(\mathcal{G}_i, \mathcal{R}_i)$ が有限であるとき $(\mathcal{G}, \mathcal{R})$ が無限の（最内）CDP 鎖を持たないならば，$Proc$ は健全であるという．

[定義 3.9]（(最内) 条件付き依存対プロセッサの完全性）$Proc(\mathcal{G}, \mathcal{R}) = \bot$ またはある $(\mathcal{G}_i, \mathcal{R}_i) \in Proc(\mathcal{G}, \mathcal{R})$ が無限の（最内）CDP 鎖をもつとき，$(\mathcal{G}, \mathcal{R})$ が無限の（最内）CDP 鎖を持つならば，$Proc$ は完全であるという．

紙面の都合上省略するが，CDP 鎖に含まれることがない辺を除いた近似グラフの概念 [4] を用いることで，より単純な依存グラフに変換するプロセッサを容易に設計することができる．

4. 関数型プログラム停止性証明のための依存対法

[1] では本論文において対象とする先行評価に基づく関数型言語 ML/ex の定義および，ML/ex プログラムから CS-TRS への変換を与えた．本節ではこの変換で得られた CS-TRS を示し (4.1 節)，CS-TRS から CDP と CDP 鎖を構成する方法を与える (4.2 節)．

4.1 関数型プログラムから変換された文脈依存項書換え系

ML/ex プログラムは，関数記号の集合，関数を定義する書換え規則の集合 $Fundef$，例外の集合の 3 項組である．プログラム例を図 1 に示す．ただし，組込み関数の定義は省略した．[1] の変換 ϕ は ML/ex のプログラム $\mathcal{P}$ を入力とし，CS-TRS $\phi(\mathcal{P})$ を出力とする．このとき，プログラム $\mathcal{P}$ の停止性は，CS-TRS $\phi(\mathcal{P})$ の最内停止性として保存される [1]．図 1 のプログラムから ϕ で変換得られる CS-TRS を図 2 に示す．ただし関数の規則の一部を省略している．ここで，関数記号 $raise$, $fire$, $isData$, $+$, $+_3$, g, g_3 の文脈 μ の値は空集合，$succ$, if, $handle$, $guard$, $select$, $+_1$, g_1 の μ 値は $\{1\}$，$+_2$ と g_2 の μ 値は $\{2\}$ である．

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

図 1 例外処理を含む ML/ex プログラム (1)
Fig. 1 Terminating ML/ex program with exception handling (1)

$$guard(tt, y) \to y,\ guard(fire(x), y) \to fire(x),$$
$$isData(succ(x)) \to isData(x),\ isData(0) \to tt,$$
$$isData(True) \to tt,\ isData(False) \to tt,$$
$$isData(fire(x)) \to fire(x),\ succ(fire(x)) \to fire(x),$$
$$if(True, y, z) \to y,\ if(False, y, z) \to z,$$
$$if(fire(x), y, z) \to fire(x),\ select(tt, x, y, z) \to x,$$
$$raise(A) \to fire(A),\ raise(B) \to fire(B),$$
$$handle(x, A, z) \to select(isData(x), x, A, z),$$
$$handle(x, B, z) \to select(isData(x), x, B, z),$$
$$select(fire(A), x, A, z) \to z,\ select(fire(B), x, B, z) \to z,$$
$$select(fire(A), x, B, z) \to fire(A),$$
$$select(fire(B), x, A, z) \to fire(B),$$
$$g(x, y) \to g_1(x, y),\ g_1(x, y) \to guard(isData(x), g_2(x, y)),$$
$$g_2(x, y) \to guard(isData(y), g_3(x, y)),$$
$$g_3(x, y) \to handle(if(le(x, y), raise(A), g(y, x)), A, 0).$$

図 2 ML/ex プログラムを変換した CS-TRS
Fig. 2 CS-TRS transformed from ML/ex program

4.2 条件付き依存対

以下では ML/ex から変換して得られる CS-TRS $\phi(\mathcal{P})$ の最内停止性の証明に焦点を合わせ，最内 CDP 問題へ変換を考える．

ML/ex から変換して得られる CS-TRS $\phi(\mathcal{P})$ では，すべての関数に対する μ の要素数は 1 以下であるため，最内リデックスは一意に定まる．しかしながら，条件付き依存対や条件付き依存対鎖を設計する際には，無限書換え系列を導く項が現在の文脈 μ では書換え不能な場所に出現することがあるため，文脈による依存対の条件部が必要となる．

まず，CDP の候補を求めるための関数 $Cand$ を定義する．$Cand$ は $\phi(\mathcal{P})$ の項を引数とする関数であり，$Cand(t)$ は項と論理式の対を要素とする集合である．直感的には，対の第 1 要素は t 中の停止しない部分項の候補であり，第 2 要素は第 1 要素の項が将来 μ 置換位置に出現して書換え対象になるための条件を表す関係論理式である．

[定義 4.1]（関数 $Cand$） $Cand(t)$ を以下のように再帰的に定義する．

(1) $t = succ(t_1)$ のとき，$Cand(t_1)$.

(2) $t = guard(t_0, f_{i+1}(t_1, \ldots, t_n)))$ のとき，$\{(u, c \wedge t_0 \twoheadrightarrow tt) \mid (u, c) \in Cand(f_{i+1}(t_1, \ldots, t_n))\}$.

(3) $t = if(t_1, t_2, t_3)$ のとき，$Cand(t_1) \cup \{(u_2, c_2 \wedge t_1 \twoheadrightarrow True) \mid (u_2, c_2) \in Cand(t_2)\} \cup \{(u_3, c_3 \wedge t_1 \twoheadrightarrow False) \mid (u_3, c_3) \in Cand(t_3)\}$.

(4) $t = handle(t_1, E, t_3)$ のとき，$Cand(t_1) \cup \{(u_3, c_3 \wedge$

$$CDP_{\phi(\mathcal{P})} = \left\{ \begin{array}{l} (\langle g^\sharp(x_1,y_1), g_1^\sharp(x_1,y_1)\rangle, true) \\ (\langle g_1^\sharp(x_2,y_2), g_2^\sharp(x_2,y_2)\rangle, isData(x_2) \rightarrow tt) \\ (\langle g_2^\sharp(x_3,y_3), g_3^\sharp(x_3,y_3)\rangle, isData(y_3) \rightarrow tt) \\ (\langle g_3^\sharp(x_4,y_4), le^\sharp(x_4,y_4)\rangle, true) \\ (\langle g_3^\sharp(x_5,y_5), g^\sharp(y_5,x_5)\rangle, le(x_5,y_5) \rightarrow False) \\ (\langle g_3^\sharp(x_6,y_6), handle^\sharp(if(le(x_6,y_6), \\ \quad raise(A), g(y_6,x_6)), A, 0)\rangle, true) \end{array} \right\}$$

図 3　図 2 の CS-TRS の CDP 集合
Fig. 3　CDP set of CS-TRS in Fig.2

$t_1 \rightarrow fire(E))\mid (u_3,c_3)\in Cand(t_3)\}\cup\{(t,true)\}$.

(5)　$t = f(t_1,\ldots,t_n)$ $(f \in \mathcal{F}_{\mathcal{B}'}\cup\mathcal{F}_{\mathcal{D}})$ のとき，$\bigcup_{1\leq i\leq n}\{(u_i, c_i\wedge_{j<i} isData(t_j)\rightarrow tt)\mid (u_i,c_i)\in Cand(t_i)\}\cup (f(t_1,\ldots,t_n), \wedge_{1\leq i\leq n} isData(t_i)\rightarrow tt)$.

(6)　$t = f_i(t_1,\ldots,t_n)$ $(f_i\in\mathcal{F}_{\mathcal{E}})$ のとき，$\{(f_i(t_1,\ldots,t_n), true)\}$.

(7)　t が上記以外のとき，$\emptyset$.

[定義 4.2]（CDP 集合）　$\phi(\mathcal{P})$ から定まる CDP の集合 $CDP_{\phi(\mathcal{P})}$ は以下のように定義される．

$$\bigcup_{l\rightarrow r\in\phi(\mathcal{P})}\{(\langle f^\sharp(t_1,\ldots,t_n), t^\sharp\rangle, c)\mid (t,c)\in Cand(r)\}$$

[例 1]　図 2 の CS-TRS の CDP は図 3 のようになる．

[定理 4.3]　CS-TRS $\phi(\mathcal{P})$ が最内停止するならば，かつそのときに限り，$\phi(\mathcal{P})$ のもとで条件付き最内 CDP 鎖となる $CDP_{\phi(\mathcal{P})}$ の要素からなる無限列が存在しない．

ここでは紙面の都合上，証明の概略を示す．SN を停止性を持つ項の集合とし，項の集合 $T^\infty_{\min}$ を $\{s\mid s\notin SN\wedge\forall t(s\rhd t\Rightarrow t\in SN)\}$ とする．また，$T^{\sharp\infty}_{\min} = \{t^\sharp\mid t\in T^\infty_{\min}\}$ とする．

[補題 4.4]　CS-TRS が停止しないなら $T^\infty_{\min}$ の要素が存在する．

[補題 4.5]　$t^\sharp\in T^{\sharp\infty}_{\min}$ ならばある規則 $l\rightarrow r$ が存在し，$t^\sharp \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} l^\sharp\sigma\in T^{\sharp\infty}_{\min}$ である．

[補題 4.6]　$t\sigma\notin SN$ である項 t と代入 σ に対して，すべての $x\in\mathcal{V}ar(t)$ および $x\sigma\trianglerighteq u'$ であるすべての u' に対して，$u'\in SN$ ならば項 t' が存在し，$(t',c)\in Cand(t)$ かつ $c\sigma$ かつ $t'\sigma\in T^\infty_{\min}$ である．

この補題は項 t のサイズに関する帰納法で証明できる．

[補題 4.7]　Range$(\sigma)\subseteqq SN$ とする．補題 4.6 より規則 $l\rightarrow r$ に対して $l\sigma\in T^\infty_{\min}$ ならば $r\trianglerighteq r'$ かつ $r'\sigma\in T^\infty_{\min}$ かつ $c\sigma$ が成立する $CDP(\langle l^\sharp, r'^\sharp\rangle, c)$ が存在する．

[補題 4.8]　$(t,c)\in Cand(r)\wedge c\sigma$ ならばある文脈 C が存在し，$r\sigma \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C[t]_\mu\sigma$ である．

この補題は $Cand$ の定義に関する帰納法で証明できる．

[証明]（定理 4.3）　無限の最内 CDP 鎖が存在するならば，かつそのときに限り CS-TRS $\phi(\mathcal{P})$ に対して無限の最内書換えが存在することを示す．

($\Leftarrow$) 無限の書換えに対して無限の最内 CDP 鎖が構成できることを示す．補題 4.4 より，停止性を持たない項に対して部分項 $t\in T^\infty_{\min}$ が存在する．$t^\sharp\in T^{\sharp\infty}_{\min}$ について，$t^\sharp \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} l^\sharp\sigma$ が成り立つような規則 $l\rightarrow r$ と代入 σ が存在する．補題 4.5 より $l^\sharp\sigma\in T^{\sharp\infty}_{\min}$ であり，$l\sigma$ のすべての真部分項は停止性を持つため Range$(\sigma)\subseteqq SN$ である．また，$r\sigma$ は停止性を持たない．補題 4.7 より $r'^\sharp\sigma\in T^{\sharp\infty}_{\min}$ となる $(\langle l^\sharp, r'^\sharp\rangle, c)\in CDP$ が得られる．これを繰り返して無限の最内 CDP 鎖が構成できる．

($\Rightarrow$) いかなる無限の最内 CDP 鎖に対しても，対応する無限の書換えが存在することを示す．$(\langle s_1^\sharp, t_1^\sharp\rangle, c_1)(\langle s_2^\sharp, t_2^\sharp\rangle, c_2)(\langle s_3^\sharp, t_3^\sharp\rangle, c_3)\cdots$ ようなとなる最内 CDP 鎖が存在すると仮定する．各 CDP に対応する書換え規則 $s_i\rightarrow C_i[t_i]$ が存在する．また，最内 CDP 鎖の定義より $t_1^\sharp\sigma_1 \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} s_2^\sharp\sigma_2, t_2^\sharp\sigma_2 \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} s_3^\sharp\sigma_3,\ldots$ を満たす代入 $\sigma_1,\sigma_2,\ldots$ が存在する．このときすべての $c_i\sigma_i$ は成立し，$s_i\sigma_i$ は最内リデックスである．従って，$s_i\sigma_i \underset{in}{\rightarrow}_{\phi(\mathcal{P})} C_i[t_i]\sigma_i$ である．このとき $(t_i,c_i)\in Cand(C_i[t_i])$ であるため，補題 4.8 より $C_i[t_i]\sigma_i \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C_i'[t_i]_\mu\sigma_i$ という書換え系列が存在する．ここで，$t_i = g(\vec{v_i})$ とおいたとき $t_i\sigma_i \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} s_{i+1}\sigma_{i+1}$ かつ $s_{i+1}\sigma_{i+1}$ が最内リデックスであるためには，$g^\sharp(\vec{v_i}) \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} g^\sharp(\vec{v_i'})\in NF$ でなければならない．このとき，$s_{i+1} = g(\vec{v_i'})$ である．それゆえ，以下のような書換えが存在し，これは無限である．

$s_1\sigma_1 \underset{in}{\rightarrow}_{\phi(\mathcal{P})} C_1[t_1]\sigma_1 \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C_1'[t_1]_\mu\sigma_1 \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C_1'[s_2]_\mu\sigma_2$
$\underset{in}{\rightarrow}_{\phi(\mathcal{P})} C_1'[C_2[t_2]]_\mu\sigma_2 \underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C_1'[C_2'[t_2']_\mu]_\mu\sigma_2$
$\underset{in}{\rightarrow}^*_{\phi(\mathcal{P})} C_1'[C_2'[s_3]_\mu]_\mu\sigma_3 \underset{in}{\rightarrow}_{\phi(\mathcal{P})}\cdots$

5.　依存グラフを用いた停止性証明

本節では CS-TRS $\phi(\mathcal{P})$ の最内停止性証明のために有効な依存対問題のプロセッサを与える．

まず本節で用いるグラフの基本操作の記法を述べる．グラフ $\mathcal{G}=(V,E)$，頂点 $v\in V$ について $delnode(\mathcal{G},v)$ は G から頂点 v とそれを含む辺を除いて得られるグラフを表す．$delnode$ は v を頂点の集合に自然に拡張できる．また，$coalesnode(\mathcal{G},v_i,v_j,v_k)$ は，二つの頂点 $v_i,v_j\in V$ を縮約させて v_k にして得られるグラフを表す．ここで，v_i もしくは v_j に接続する辺は v_k に接続させるが，辺 (v_k,v_k) は導入しない．

$sccomp$ はグラフを強連結成分に分割する関数である．すなわち，$sccomp(\mathcal{G}) = \{\mathcal{G}_1,\ldots,\mathcal{G}_n\}$．ここで，$\mathcal{G}_1,\ldots,\mathcal{G}_n$ は $\mathcal{G}$ の分割，各 $\mathcal{G}_i$ は強連結グラフ，かつ，相異なる $\mathcal{G}_i$ と $\mathcal{G}_j$ は非連結である．

以下では CDP プロセッサに用いられる依存対問題のいくつかの性質を示す．ここでは最内書換えの場合のみ議論するが，通常の書換えにも容易に拡張可能である．

[定理 5.1]（無限条件付き依存対鎖のない条件付き依存対グラフ）最内 CDP 問題 $(\mathcal{G},\mathcal{R})$ に対して $\mathcal{G}$ 上に $\mathcal{R}$ の無限の最内 CDP 鎖が存在しないとき，$Proc(\mathcal{G},\mathcal{R})=\emptyset$ とする．$Proc$ は健全かつ完全である．

次の定理は，頂点の条件付き依存対 $(\langle s^\sharp, t^\sharp\rangle, c)$ の条件 c が $\underset{in}{\rightarrow}^*_{\mathcal{R}}$ 充足不能であればその頂点を取り除けることを表す．

[定理 5.2]（頂点の削除）　最内 CDP 問題 $((V,E),\mathcal{R})$ において，c が $\underset{in}{\rightarrow}^*_{\mathcal{R}}$ 充足不能な頂点を $v=(\langle s^\sharp, t^\sharp\rangle, c)\in V$ とする．

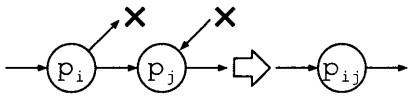

図 4　頂点の縮約
Fig. 4　coalesing of vertices

$Proc((V,E),\mathcal{R}) = (delnode((V,E),\{v\}),\mathcal{R})$ とすると，$Proc$ は健全かつ完全である．

次の定理は，辺の始点と終点の CDP が CDP 鎖となるような代入が存在しなければその辺を取り除けることを表す．

[定理 5.3]（辺の削除）　最内 CDP 問題 $((V,E),\mathcal{R})$ において $p = ((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j)) \in E$ とする．$t_i^\sharp\sigma_i \underset{in}{\rightarrow}^*_{\mathcal{R}} s_j^\sharp\sigma_j$，$\langle \underset{in}{\rightarrow}_{\mathcal{R}}, \sigma_i\rangle \vdash c_i$ かつ $\langle \underset{in}{\rightarrow}_{\mathcal{R}}, \sigma_j\rangle \vdash c_j$ を満たす代入 σ_i, σ_j が存在しないならば，$Proc((V,E),\mathcal{R}) = ((V, E\setminus\{p\}),\mathcal{R})$ とする．このとき $Proc$ は健全かつ完全である．

定理 5.3 の $Proc$ によって $root(t_i^\sharp) \neq root(s_j^\sharp)$ であるような辺 $((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j))$ は取り除くことができる．

[定理 5.4]（強連結成分への分割）　$(\mathcal{G},\mathcal{R})$ を最内 CDP 問題とする．$sccomp(\mathcal{G}) = \{\mathcal{G}_1,\ldots,\mathcal{G}_n\}$ であるとき，$Proc(\mathcal{G},\mathcal{R}) = \{(\mathcal{G}_1,\mathcal{R}),\ldots,(\mathcal{G}_n,\mathcal{R})\}$ とすると，$Proc$ は健全かつ完全である．

次の定理は，図 4 のように，分岐のない頂点と合流のない頂点が隣接する場合は頂点を一つに縮約し，もとの頂点を取り除くことができることを表す．

[定理 5.5]（頂点の縮約）　CDP 問題 $((V,E),\mathcal{R})$ において，$p_i = (\langle s_i^\sharp, t_i^\sharp\rangle, c_i), p_j = (\langle s_j^\sharp, t_j^\sharp\rangle, c_j) \in V$，$(p_i,p_j) \in E$ かつ $t_i^\sharp\sigma_i \underset{in}{\rightarrow}^*_{\mathcal{R}} s_j^\sharp\sigma_j$ が成り立つような代入 σ_i,σ_j が存在するとする．$p'_j \neq p_j$ ならば $(p_i,p'_j) \notin E$ かつ $p'_i \neq p_i$ ならば $(p'_i,p_j) \notin E$ であるとき，$Proc(((V,E),\mathcal{R})) = (coalesnode((V,E),(\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j), (\langle s_i^\sharp\sigma_i, t_j^\sharp\sigma_j\rangle, c_i\sigma_i \wedge c_j\sigma_j)),\mathcal{R})$ とすると，$Proc$ は健全かつ完全である．

[証明]　$(V',E') = coalesnode((V,E),(\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j), (\langle s_i^\sharp\sigma_i, t_j^\sharp\sigma_j\rangle, c_i\sigma_i \wedge c_j\sigma_j))$ とする．

- (V,E) 上に無限の CDP 鎖が存在し，$((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j))$ が含まれる場合．

以下のような鎖が存在したとする．

$$((\langle s_l^\sharp, t_l^\sharp\rangle, c_l), (\langle s_i^\sharp, t_i^\sharp\rangle, c_i)),$$
$$((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j)),$$
$$((\langle s_j^\sharp, t_j^\sharp\rangle, c_j), (\langle s_m^\sharp, t_m^\sharp\rangle, c_m))$$

$Proc$ によって以下のような鎖が代わりに現れる．

$$((\langle s_l^\sharp, t_l^\sharp\rangle, c_l), (\langle s_i^\sharp\sigma_i, t_j^\sharp\sigma_j\rangle, (c_i\sigma_i \wedge c_j\sigma_j)),$$
$$((\langle s_i^\sharp\sigma_i, t_j^\sharp\sigma_j\rangle, c_i\sigma_i \wedge c_j\sigma_i), (\langle s_m^\sharp, t_m^\sharp\rangle, c_m))$$

この 2 つの系列が鎖となる条件は同じである．

- (V,E) 上に無限の CDP 鎖が存在するが，$((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j))$ が含まれない場合．

$Proc$ によって無限 CDP 鎖は変化しない．

- (V,E) 上に無限 CDP 鎖が存在しない場合．

$Proc$ によって無限 CDP 鎖が出現することはない．

よって $Proc$ は健全かつ完全である．

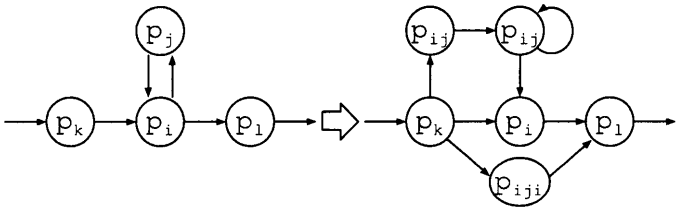

図 5　隣接閉路の展開
Fig. 5　expansion of adjacent circuit

[例 2]　条件付き項書換え系 $\phi(\mathcal{P})$ において，*if*, *handle*, *raise* 以外の被定義記号 f に対して以下の条件付き依存対が存在し，最内 CDP 鎖となる．

$$(\langle f^\sharp(x_1,\ldots,x_n), f_1^\sharp(x_1,\ldots,x_n)\rangle, true)$$
$$(\langle f_1^\sharp(x_1,\ldots,x_n), f_2^\sharp(x_1,\ldots,x_n)\rangle, isData(x_1) \to tt)$$
$$\vdots$$
$$(\langle f_n^\sharp(x_1,\ldots,x_n), f_{n+1}^\sharp(x_1,\ldots,x_n)\rangle, isData(x_n) \to tt)$$

この最内 CDP 鎖は条件付き依存対プロセッサを繰り返し適用することで以下の 1 個の条件付き依存対 $(\langle f^\sharp(x_1,\ldots,x_n), f_{n+1}^\sharp(x_1,\ldots,x_n)\theta_1\cdots\theta_n\rangle, c \wedge isData(x_1) \twoheadrightarrow tt \wedge \cdots \wedge (isData(x_n) \to tt)\theta_1\cdots\theta_n)$ となる．

閉路を 1 回まわる条件は成立しても 2 回まわると成立しなくなる場合がある．p_i を始点とする閉路があり，この頂点以外の閉路上の頂点では分岐，合流がないものとする．このとき，定理 5.5 を適用して図 5 の左側のグラフのように閉路上の途中の頂点を一つの頂点 p_j にまとめることができる．次の定理は p_i と p_j のような隣接閉路を図 5 の右側のグラフのように展開できることを表す．

[定理 5.6]（隣接閉路の展開）　(V,E) が $\phi(\mathcal{P})$ の CDP グラフである CDP 問題 $((V,E),\mathcal{R})$ において，$\{((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j)), ((\langle s_j^\sharp, t_j^\sharp\rangle, c_j), (\langle s_i^\sharp, t_i^\sharp\rangle, c_i))\} \subseteqq E$ かつ，$t_i^\sharp\sigma_i \underset{in}{\rightarrow}^*_{\mathcal{R}} s_j^\sharp\sigma_j, t_j^\sharp\sigma_j \underset{in}{\rightarrow}^*_{\mathcal{R}} s_i^\sharp\sigma_i$ が成り立つような代入 σ_i,σ_j が存在し，$(\langle s_j^\sharp, t_j^\sharp\rangle, c_j)$ において分岐，合流がないとき，$Proc(((V,E),\mathcal{R})) =$

$((V \setminus \{(\langle s_j^\sharp, t_j^\sharp\rangle, c_j))\} \cup \{(\langle s_i^\sharp, s_i^\sharp\rangle, (c_i \wedge c_j)\sigma_i\sigma_j), (\langle s_i^\sharp, t_j^\sharp\rangle, (c_i \wedge c_j)\sigma_i)\},$

$E \setminus \{((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_j^\sharp, t_j^\sharp\rangle, c_j)), ((\langle s_j^\sharp, t_j^\sharp\rangle, c_i), (\langle s_i^\sharp, t_i^\sharp\rangle, c_i))\} \cup$
$\{((\langle s_k^\sharp, t_k^\sharp\rangle, c_k), (\langle s_i^\sharp, s_i^\sharp\rangle, (c_i \wedge c_j)\sigma_i\sigma_j)),$
$((\langle s_i^\sharp, s_i^\sharp\rangle, (c_i \wedge c_j)\sigma_i\sigma_j), (\langle s_l^\sharp, t_l^\sharp\rangle, c_l)),$
$((\langle s_k^\sharp, t_k^\sharp\rangle, c_k), (\langle s_i^\sharp, t_j^\sharp\rangle, (c_i \wedge c_j)\sigma_i)),$
$((\langle s_i^\sharp, t_j^\sharp\rangle, (c_i \wedge c_j)\sigma_i), (\langle s_i^\sharp, t_i^\sharp\rangle, c_i))\}$
$| \{((\langle s_k^\sharp, t_k^\sharp\rangle, c_k), (\langle s_i^\sharp, t_i^\sharp\rangle, c_i)), ((\langle s_i^\sharp, t_i^\sharp\rangle, c_i), (\langle s_l^\sharp, t_l^\sharp\rangle, c_l))\} \subseteqq E, \mathcal{R})$

とすると，$Proc$ は健全かつ完全である．

なお，既存の DP プロセッサ [4] の技法の多くは CDP プロセッサにも導入可能である．

6.　CDP を用いた停止性証明例

本節では CDP を用いたプログラムの停止性証明の例を示す．

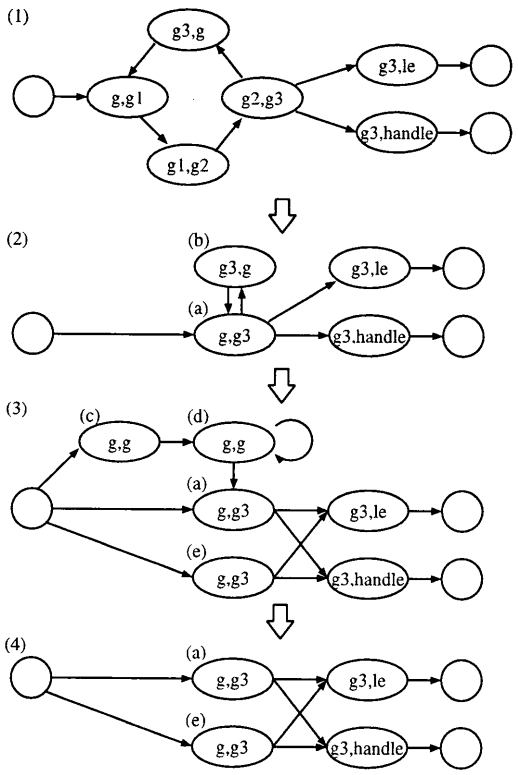

図 6　グラフの変換
Fig. 6　transformation of graph

6.1 例

定理 4.3 より図 2 の CS-TRS の停止性は最内 CDP 鎖の存在性に置き換えることができる．また，この存在性は補題 3.6 より図 3 の条件付き依存対 $CDP_{\phi(\mathcal{P})}$ を頂点とする完全グラフを $\mathcal{G}_c$ とするとき，$(\mathcal{G}_c, \mathcal{R})$ を入力とする最内 CDP 問題の有限性を調べればよい．$(\mathcal{G}_c, \mathcal{R})$ に定理 5.3 のプロセッサを適用すると図 6 の (1) のグラフが得られる．(1) に定理 5.5 のプロセッサを適用すると (2) のグラフが得られる．(2) に定理 5.6 のプロセッサを適用すると (3) のグラフが得られる．頂点 (a),(b),(c),(d),(e) は以下のとおりである．

(a): $(\langle g^\sharp(x_7, y_7), g_3^\sharp(x_7, y_7)\rangle, isData(x_7) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_7) \rightarrow\!\!\!\!\!\rightarrow tt)$.

(b): $(\langle g_3^\sharp(x_8, y_8), g^\sharp(y_8, x_8)\rangle, le(x_8, y_8) \rightarrow\!\!\!\!\!\rightarrow False)$.

(c): $(\langle g^\sharp(x_9, y_9), g^\sharp(y_9, x_9)\rangle, isData(x_9) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_9) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_9, y_9) \rightarrow\!\!\!\!\!\rightarrow False)$.

(d): $(\langle g^\sharp(x_{10}, y_{10}), g^\sharp(y_{10}, x_{10})\rangle, isData(x_{10}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_{10}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_{10}, y_{10}) \rightarrow\!\!\!\!\!\rightarrow False)$.

(e): $(\langle g^\sharp(x_{11}, y_{11}), g_3^\sharp(y_{11}, x_{11})\rangle, isData(x_{11}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_{11}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_{11}, y_{11}) \rightarrow\!\!\!\!\!\rightarrow False)$.

ここで，経路 (c)(d) が CDP 鎖となる条件は $\sigma = \{x_9 := x_{12}, y_9 := y_{12}, x_{10} := y_{12}, y_{10} := x_{12}\}$ とすると，$(isData(x_9) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_9) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_9, y_9) \rightarrow\!\!\!\!\!\rightarrow False \wedge isData(x_{10}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_{10}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_{10}, y_{10}) \rightarrow\!\!\!\!\!\rightarrow False)\sigma$ である．$isData(x_{12}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge isData(y_{12}) \rightarrow\!\!\!\!\!\rightarrow tt \wedge le(x_{12}, y_{12}) \rightarrow\!\!\!\!\!\rightarrow False \wedge le(y_{12}, x_{12}) \rightarrow\!\!\!\!\!\rightarrow False$ は充足不能であるため，定理 5.3 のプロセッサを用いて，辺 (c)(d) は削除できる．同様に辺 (d)(d) も削除できる．また，頂点 (c) を始点とする辺と頂点 (d) を終点とする辺が存在しなくなるため，定理 5.4 のプロセッサを用いて削除でき，(4) のグラフになる．この (部分) グラフには無限長の経路は存在しない．

7. 関連研究

[6] では組み込み意味論で評価される制約の付いた書換え規則を用いる制約付き項書換え系の停止性を証明するために graph-handling DP framework を提案している．依存対に条件部が付加されているが本論文とは条件の形式が異なる．また，本論文と同様に依存グラフを導入した GDP 問題を扱っている．その中では制約を利用したグラフの辺を削除するプロセッサを与えているが，本論文で導入したような新たな頂点を加えるプロセッサは定義されていない．

8. おわりに

本論文では例外処理を持つ関数型プログラムから変換された CS-TRS の新しい停止性証明法を提案した．条件付き依存対と条件付き依存対鎖の定義を与え，例外処理を持つ関数型プログラムから変換された CS-TRS の最内停止性と無限の最内条件付き依存対鎖の非存在が一致することを証明した．また，DP プロセッサ [4] と GDP プロセッサ [6] を拡張し，例外処理を含む関数型プログラム停止性証明に有効な条件付き依存対問題のプロセッサの定義を与えた．本手法による停止性証明器を実現し，有用性を示すことが今後の課題である．

文　献

[1] 濱口　毅，馬場正貴，酒井正彦，阿草清滋，“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，” 情報処理学会論文誌 プログラミング，vol.4, no.2, pp.13–30, 2011.

[2] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke, “Automated termination proofs with aprove,” Proceedings of the 15th International Conference on Rewriting Techniques and Applications (RTA-04)(Lecture Notes in Computer Science 3091), pp.210–220, Aachen, Germany, 2004.

[3] B. Alarcón, R. Gutiérrez, J. Iborra, and S. Lucas, “Proving termination of context-sensitive rewriting with mu-term,” Proceedings of the Sixth Spanish Conference on Programming and Computer Languages, PROLE 2006(Electronic Notes in Theoretical Computer Science, 188), pp.105–115, 2007.

[4] J. Giesl, R. Thiemann, and P. Schneider-Kamp, “The dependency pair framework: Combining techniquies for automated termination proofs,” Proceedings of 11th LPAR,(LNAI 3452), pp.301–331, 2005.

[5] J. Giesl, S. Swiderski, P. Schneider-Kamp, and R. Thiemann, “Automated termination analysis for haskell : From term rewriting to programming languages,” Proceedings of Term Rewriting and Applications 17th International Conference RTA 2006(Lecture Notes in Computer Science 4098), pp.297–312, Seattle, USA, 2006.

[6] T. Sakata, N. Nishida, and T. Sakabe, “On proving termination of constrained term rewrite systems by eliminating edges from dependency graphs,” Proceedings of the 20th International Workshop on Functional and (Constraint) Logic Programming (WFLP 2011) in Vol. 6816 of Lecture Notes in Computer Science, pp.138–155, 2011.

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