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
SS2009-41 (2009-12)

条件付き等式の変換に基づくプログラム生成

長島　正憲† 　酒井　正彦† 　坂部　俊樹† 　西田　直樹† 　草刈圭一朗†

† 名古屋大学大学院情報科学研究科　〒464-8603 名古屋市千種区不老町
E-mail: † nagashima@sakabe.i.is.nagoya-u.ac.jp, {sakai,sakabe,nishida,kusakari}@is.nagoya-u.ac.jp

あらまし　以前に我々が提案したプログラム生成系 *GeneSys* は，一階述語論理式で表現された仕様から実行可能なプログラムを生成するが，変換過程の論理式が複雑になりやすい欠点がある．本稿では，仕様の表現方法を条件付き等式（ホーン節）に制限し，単純かつ直観的なプログラム生成系の構築を目指す．その上で，プログラム生成のいくつかの実例を挙げる．

キーワード　プログラム生成，条件付き等式，ホーン節，項書換え系

Program Generation Based on Transformation of Conditional Equations

Masanori NAGASHIMA†, Masahiko SAKAI†, Toshiki SAKABE†,

Naoki NISHIDA†, and Keiichirou KUSAKARI†

† Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8603 Japan
E-mail: † nagashima@sakabe.i.is.nagoya-u.ac.jp, {sakai,sakabe,nishida,kusakari}@is.nagoya-u.ac.jp

Abstract　Program-generation system *GeneSys*, which we have ever proposed, generates executable programs from first order equational specifications. The system is, however, apt to create complicated formulas during transformation. Instead, in this paper, we use conditional equations (Horn clauses) as specifications to try constructing a more simple and more intuitive system. On that basis, we give examples of program generation.

Key words　program generation, conditional equation, Horn clause, term rewriting system

1. はじめに

プログラムが満たすべき性質を記述した仕様から実行可能なプログラムを作成する作業は，人的リソースが不可欠で自動化が困難であると一般に考えられている．一方，既に実行可能なプログラムを効率のよい，あるいは別の目的を持つプログラムに変換する手法はいくつか提案されており，複数の関数の組み合わせを単一の関数に融合することで効率の良いプログラムを生成する融合変換 [8] や，関数の逆写像を計算するプログラムを生成する変換 [4] などが知られている．

プログラム生成系 *GeneSys* [2], [3] は，前述の変換では対象としていない，(実行不可能な) 仕様から実行可能なプログラムを得るための変換を提案しており，実例はまだ少なく完全に自動化できるわけではないが，プログラム生成の成功例がいくつか挙げられている．また関連研究として *GeneSys* の変換規則の適用戦略が提案されており [1]，その戦略で融合変換の一手法である Deforestation [8] を模倣できることが示されている．しかしながら，*GeneSys* の変換対象は一階述語論理式であり，仕様を変換していく過程で論理記号の入れ子構造が深くなることが多い．また論理和や存在限量子は論理積や全称限量子に比較して取扱いがかなり難しいことが次第に明らかになってきた．

本稿では，変換対象とする論理式を条件付き等式（ホーン節）に限定し，ホーン節に閉じた変換をするように *GeneSys* の変換規則を再構築する．その上で新しい変換規則によるプログラム生成例をいくつか挙げ，より単純で直感的な変換になっていることを確認する．

2. 準　　備

本節では，基本的な記法や概念を定義する．詳しくは文献 [5], [6] などを参照して頂きたい．

$\mathcal{X}$ を**変数**の（可算無限）集合，$\mathcal{F}$ を**関数記号**の集合，$\mathcal{K}$ を型の集合とする．変数や関数記号にはそれぞれ（引数や返値の）**型**が対応付けられているものとする．変数 $x \in \mathcal{X}$ の型 $\kappa \in \mathcal{K}$ を明記する場合 $x_{:\kappa}$ と書く．同様に引数の型 $\kappa_1 \cdots \kappa_n \in \mathcal{K}$，返値の型 $\kappa \in \mathcal{K}$ を持つ関数記号 $f \in \mathcal{F}$ は $f : \kappa_1 * \cdots * \kappa_n \to \kappa$ と書く．特に f が**定数記号** ($n = 0$) の場合は $f : \kappa$ と書く．

$\mathcal{F}, \mathcal{X}$ 上の（**型付き**）**項**は，次のように再帰的に定義される．

（1）　型 κ の変数 $x \in \mathcal{X}$ は型 κ の項である．

（2） 型がそれぞれ $\kappa_1,\ldots,\kappa_n$ の項 $t_1,\ldots,t_n$ と関数記号 $f:\kappa_1*\cdots*\kappa_n\to\kappa\in\mathcal{F}$ に対して $f(t_1,\ldots,t_n)$ は型 κ の項である．ただし，項 $f()$ は f と略記する．

項 t の型 κ を明記する場合 $t_{:\kappa}$ と書く．$\mathcal{F},\mathcal{X}$ 上のすべての項の集合を $\mathcal{T}(\mathcal{F},\mathcal{X})$ で，型 κ を持つすべての項の集合を $\mathcal{T}(\mathcal{F},\mathcal{X})_{:\kappa}$ でそれぞれ表す．変数を含まない項を**基底項**といい，その集合 $\mathcal{T}(\mathcal{F},\emptyset)$ を $\mathcal{T}(\mathcal{F})$ と略記する．2つの項 s,t が同一であるとき $s=t$ と書く．

本稿で取り扱う論理記号は $\wedge$（**論理積**），$\Leftarrow$（**含意**），$\forall$（**全称限量子**）の3種類，述語記号は $\approx$（**等号**），$\not\approx$（**不等号**）の2種類である．つまり，同一の型を持つ項 s,t によってできる**等式** $s\approx t$，**不等式** $s\not\approx t$ のみが**原子論理式**である．一階述語論理や項書換え系においては，等式 $s\approx t$ と $t\approx s$ の解釈は一致するので，本稿ではこれらを同一の等式とみなす（不等式についても同様）．さらに，$\wedge$ に関する交換則や結合則，$\forall$ に束縛される変数の名前替え（α 変換），連続した $\forall$ の束縛順序の入れ替えを常に許すものとする．これらの意味で2つの論理式 U,V が同一であるとき $U\equiv V$ と書く．

変数に自身と同一の型を持つ項を対応付ける写像 $\sigma:\mathcal{X}\to\mathcal{T}(\mathcal{F},\mathcal{X})$ は，$x\neq\sigma(x)$ となる x が有限個であるとき**代入**であるという．代入 σ は $\{x\mapsto\sigma(x)\mid x\neq\sigma(x)\}$ の形式で表現する．代入 σ は項上および論理式上への写像に自然に拡張される．すなわち，$\sigma(t),\sigma(U)$ はそれぞれ項 t，論理式 U 中の各変数 x を項 $\sigma(x)$ に置き換えて得られる項である．ただし，$\forall$ で束縛される変数は名前替えによって σ の定義域や値域に含まれない変数になっているものとする．$\sigma(t),\sigma(U)$ はそれぞれ $t\sigma,U\sigma$ と略記する．

特別な定数記号 $\square$ をちょうど1個含む項 C を**項文脈**という．項文脈 C の $\square$ を同一の型を持つ項 t で置き換えて得られる項を $C[t]$ と表す．同様に，特別な述語記号 $\diamond$ をちょうど1個含む論理式 Γ をを**論理式文脈**といい，その $\diamond$ を論理式 U に置き換えて得られる論理式を $\Gamma\langle U\rangle$ と表す．

次の条件を満たす3項組 $(l,r,Cond)$ を（**条件付き**）**書換え規則**といい，$l\to r\Leftarrow Cond$ と書く．

- $l,r\in\mathcal{T}(\mathcal{F},\mathcal{X})$ かつ $l\notin\mathcal{X}$
- $Cond$ は条件部と呼ばれ，true または $s_1\approx t_1\wedge\cdots\wedge s_n\approx t_n$ $(n>0)$ の形式をしている．

条件部が true である規則 $l\to r\Leftarrow$ true は $l\to r$ と略記する．条件付き書換え規則の有限集合を**条件付き項書換え系**(CTRS)という．CTRS $\mathcal{R}$ による n レベルの書換え関係 $\xrightarrow[n]{}_{\mathcal{R}}$ は次のように再帰的に定義される．

$$\xrightarrow[0]{}_{\mathcal{R}}=\emptyset$$

$$\xrightarrow[n+1]{}_{\mathcal{R}}=\left\{(C[l\sigma],C[r\sigma])\mid l\to r\Leftarrow Cond\in\mathcal{R},\ \forall s_i\approx t_i.\ s_i\sigma\xleftrightarrow[n]{*}_{\mathcal{R}}t_i\sigma\right\}$$

ただし，$\xleftrightarrow[n]{*}_{\mathcal{R}}$ は $\xrightarrow[n]{}_{\mathcal{R}}$ の反射・推移・対称閉包である．$\mathcal{R}$ による書換え関係は $\to_{\mathcal{R}}=\bigcup_{n\geq 0}\xrightarrow[n]{}_{\mathcal{R}}$ と定義される．

3. ホーン節の変換に基づくプログラム生成系

本稿で新たに提案するプログラム生成系は条件付き等式（ホーン節）を変換の対象とする．ホーン節はプログラミング言語 Prolog などでよく知られている．

[定義 3.1]（**ホーン節**） 次の条件を満たす論理式 $Clause\equiv e\Leftarrow Cond$ をホーン節という．

- e は原始論理式（本稿では等式）または false
- $Cond$ は原始論理式の連言 $e_1\wedge\cdots\wedge e_n$ または true

なお，すべての変数は（表記上は省略される）冠頭の全称限量子によって束縛されているものとする．すなわち自由変数は存在しない．ホーン節の集合は，節の論理積とみなすことができる． □

GeneSys では場合分けが網羅すべき対象を被覆集合 [7] によって定義している．本稿でもこの概念を引き続き利用する．

[定義 3.2]（**被覆集合**） $\mathcal{R}$ を CTRS，S を型 κ を持つ項の集合とする．型 κ を持つすべての基底項 $t\in\mathcal{T}(\mathcal{F})_{:\kappa}$ に対して，ある項 $s\in S$ とある代入 σ が存在して $t\xleftrightarrow{*}_{\mathcal{R}}s\sigma$ を満たすとき，S を型 κ の被覆集合であるという．なお，型 κ のすべての被覆集合からなる族を $\mathcal{CS}_{:\kappa}$ で表す． □

以上の準備の下で *GeneSys* の変換規則を再構築する．

[定義 3.3]（プログラム生成系）

本稿におけるプログラム生成系は，

- 仕様 $\mathcal{E}$：ホーン節の集合
- ライブラリ $\mathcal{R}$：CTRS（実行可能な既存プログラム）

から出発し，以下に列挙されている変換規則に従って，$\mathcal{R}$ の下で $\mathcal{E}$ 中のホーン節を変換する．この変換の1ステップを $\mathcal{E}\Rightarrow_{\mathcal{R}}\mathcal{E}'$ と表記する．

(Expansion)
$$\frac{\mathcal{E}\cup\{Clause\}}{\mathcal{E}\cup\{\cdots,Clause\{x_{:\kappa}\mapsto t_i\},\cdots\}}\ \text{if}\ \{\ldots,t_i,\ldots\}\in\mathcal{CS}_{:\kappa}$$

(Simplification)
$$\frac{\mathcal{E}\cup\{e\Leftarrow Cond\langle C[l\sigma]\approx t\wedge(Cond')\sigma\rangle\}}{\mathcal{E}\cup\{e\Leftarrow Cond\langle C[r\sigma]\approx t\wedge(Cond')\sigma\rangle\}}\ \text{if}\ l\to r\Leftarrow Cond'\in\mathcal{R}$$

(Deduction)
$$\frac{\mathcal{E}\cup\{e\Leftarrow Cond\langle Cond'\rangle\}}{\mathcal{E}\cup\{e\Leftarrow Cond\langle e'\rangle\}}\ \text{if}\ e'\Leftarrow Cond'\in\mathcal{E}$$

(Decomposition)
$$\frac{\mathcal{E}\cup\{e\Leftarrow Cond\langle C[s]\approx t\rangle\}}{\mathcal{E}\cup\{e\Leftarrow Cond\langle C[x]\approx t\wedge x\approx s\rangle\}}\ \text{if}\ x\ \text{is fresh}$$

(Composition)
$$\frac{\mathcal{E}\cup\{e\Leftarrow Cond\langle x\approx t\rangle\}}{\mathcal{E}\cup\{(e\Leftarrow Cond\langle\text{true}\rangle)\{x\mapsto t\}\}}$$

(Introduction)
$$\frac{\mathcal{E}}{\mathcal{E}\cup\{Clause\}}$$

(Case-Division)
$$\frac{\mathcal{E}\cup\{e\Leftarrow Cond\}}{\mathcal{E}\cup\{e\Leftarrow Cond\wedge s\approx t,\ e\Leftarrow Cond\wedge s\not\approx t\}}$$

□

プログラム生成が成功するのは，変換系列 $\mathcal{E}_0(\equiv \mathcal{E}) \Rightarrow_{\mathcal{R}} \cdots \Rightarrow_{\mathcal{R}} \mathcal{E}_n$ によって導かれる $\mathcal{E}_n$ が実行可能な定義になっている場合である．

変換規則を実際に適用する様子は次節で紹介するが，各規則の働きはおおよそ次のようになる．

Expansion： 被覆集合による変数の場合分け
Simplification： ライブラリによる計算
Deduction： 他規則を適用後に再帰定義を作る変換
Decomposition： 合成関数の紐解き
Composition： 代入による変数の消去
Introduction： 補題の追加
Case-Division： ある等式が成立するか否かによる場合分け

なお，$\mathcal{R}$ の下での規則 Expansion, Simplification, Deduction, Decomposition, Composition, Introduction, Case-Division による1ステップの変換をそれぞれ $\Rightarrow^{\mathsf{Exp}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Sim}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Ded}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Dec}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Com}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Int}}_{\mathcal{R}}$, $\Rightarrow^{\mathsf{Cas}}_{\mathcal{R}}$ と表す．

4. プログラムの生成例

本節では，定義 3.3 の変換規則によるプログラム生成例をいくつか挙げる．

[例 4.1]（自然数の減算を実行するプログラム Sub の生成例）
ライブラリとして，自然数の加算を実行するプログラム Add を実行可能な CTRS が与えられている．仕様 $\mathcal{E}_{\mathsf{Sub}}$ とライブラリ $\mathcal{R}_{\mathsf{Add}}$ は以下で定義される．

$$\mathcal{E}_{\mathsf{Sub}} = \left\{ \mathsf{Sub}(x,y) \approx z \Leftarrow \mathsf{Add}(z,y) \approx x \ \cdots(1) \right\}$$

$$\mathcal{R}_{\mathsf{Add}} = \left\{ \begin{array}{l} \mathsf{Add}(x,0) \to x, \\ \mathsf{Add}(x,\mathsf{s}(y)) \to \mathsf{s}(\mathsf{Add}(x,y)) \end{array} \right\}$$

$$\left(\begin{array}{l} 0 : \mathbb{N}, \quad \mathsf{s} : \mathbb{N} \to \mathbb{N}, \\ \mathsf{Add} : \mathbb{N} * \mathbb{N} \to \mathbb{N}, \quad \mathsf{Sub} : \mathbb{N} * \mathbb{N} \to \mathbb{N} \end{array} \right)$$

$\mathcal{R}_{\mathsf{Add}}$ の下での $\mathcal{E}_{\mathsf{Sub}}$ の変換系列（の1つ）を以下に示す．

$\mathcal{E}_{\mathsf{Sub}}$

$$\overset{*}{\Rightarrow}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{Add}}} \left\{ \begin{array}{l} \mathsf{Sub}(x,\underline{0}) \approx z \Leftarrow \mathsf{Add}(z,\underline{0}) \approx x, \\ \mathsf{Sub}(x,\underline{\mathsf{s}(y)}) \approx z \Leftarrow \mathsf{Add}(z,\underline{\mathsf{s}(y)}) \approx x, \ (1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Sim}}_{\mathcal{R}_{\mathsf{Add}}} \left\{ \begin{array}{l} \mathsf{Sub}(x,0) \approx z \Leftarrow \underline{z} \approx x, \\ \mathsf{Sub}(x,\mathsf{s}(y)) \approx z \Leftarrow \underline{\mathsf{s}(\mathsf{Add}(z,y))} \approx x, \ (1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Dec}}_{\mathcal{R}_{\mathsf{Add}}} \left\{ \begin{array}{l} \mathsf{Sub}(x,0) \approx z \Leftarrow z \approx x, \\ \mathsf{Sub}(x,\mathsf{s}(y)) \approx z \\ \quad \Leftarrow \mathsf{s}(\underline{w}) \approx x \wedge \underline{w \approx \mathsf{Add}(z,y)}, \ (1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Ded}}_{\mathcal{R}_{\mathsf{Add}}} \left\{ \begin{array}{l} \mathsf{Sub}(x,0) \approx z \Leftarrow z \approx x, \\ \mathsf{Sub}(x,\mathsf{s}(y)) \approx z \\ \quad \Leftarrow \mathsf{s}(w) \approx x \wedge \underline{\mathsf{Sub}(w,y) \approx z}, \ (1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Com}}_{\mathcal{R}_{\mathsf{Add}}} \left\{ \begin{array}{l} \mathsf{Sub}(\underline{z},0) \approx z, \\ \mathsf{Sub}(\underline{\mathsf{s}(w)},\mathsf{s}(y)) \approx \underline{\mathsf{Sub}(w,y)}, \ (1) \end{array} \right\}$$

この変換により Sub を実行可能な CTRS

$$\mathcal{R}_{\mathsf{Sub}} = \left\{ \begin{array}{l} \mathsf{Sub}(z,0) \to z, \\ \mathsf{Sub}(\mathsf{s}(w),\mathsf{s}(y)) \to \mathsf{Sub}(w,y) \end{array} \right\}$$

を得る．この $\mathcal{R}_{\mathsf{Sub}}$ は元の仕様 $\mathcal{E}_{\mathsf{Sub}}$ を確かに満たしている．

□

例 4.1 では補題を追加するための規則 Introduction を使っていない．しかし，定理自動証明の過程でよく見られるように，本稿のプログラム生成においても補題が必要になる事例は少なくない．次の例では，Introduction 規則が使用される．

[例 4.2]（自然数の除算を実行するプログラム Quot の生成例）
ライブラリとして，自然数の乗算を実行するプログラム Mult を実行可能な CTRS が与えられている．仕様 $\mathcal{E}_{\mathsf{Quot}}$ とライブラリ $\mathcal{R}_{\mathsf{Mult}}$ は以下で定義される．

$$\mathcal{E}_{\mathsf{Quot}} = \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx z \\ \quad \Leftarrow \mathsf{Mult}(z,y) \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true} \ \cdots(2) \end{array} \right\}$$

$$\mathcal{R}_{\mathsf{Mult}} = \left\{ \begin{array}{l} \mathsf{Mult}(x,0) \to 0, \quad \mathsf{Mult}(0,y) \to 0, \\ \mathsf{Mult}(\mathsf{s}(x),\mathsf{s}(y)) \\ \quad \to \mathsf{s}(\mathsf{Add}(\mathsf{Mult}(x,\mathsf{s}(y)),y)), \\ \mathsf{Add}(x,0) \to x, \\ \mathsf{Add}(x,\mathsf{s}(y)) \to \mathsf{s}(\mathsf{Add}(x,y)), \\ \mathsf{Gt}(0,y) \to \mathsf{false}, \quad \mathsf{Gt}(\mathsf{s}(x),0) \to \mathsf{true}, \\ \mathsf{Gt}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{Gt}(x,y) \end{array} \right\}$$

$$\left(\begin{array}{l} \mathsf{true} : \mathbb{B}, \quad \mathsf{false} : \mathbb{B}, \quad 0 : \mathbb{N}, \quad \mathsf{s} : \mathbb{N} \to \mathbb{N}, \\ \mathsf{Gt} : \mathbb{N} * \mathbb{N} \to \mathbb{B}, \quad \mathsf{Add} : \mathbb{N} * \mathbb{N} \to \mathbb{N}, \\ \mathsf{Mult} : \mathbb{N} * \mathbb{N} \to \mathbb{N}, \quad \mathsf{Quot} : \mathbb{N} * \mathbb{N} \to \mathbb{N} \end{array} \right)$$

$\mathcal{R}_{\mathsf{Mult}}$ の下での $\mathcal{E}_{\mathsf{Quot}}$ の変換系列（の1つ）を以下に示す．

$\mathcal{E}_{\mathsf{Quot}}$

$$\overset{*}{\Rightarrow}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \\ \quad \Leftarrow \mathsf{Mult}(0,y) \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,y) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{Mult}(\mathsf{s}(z),y) \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \ (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \\ \quad \Leftarrow \mathsf{Mult}(0,y) \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,0) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{Mult}(\mathsf{s}(z),0) \approx x \wedge \mathsf{Gt}(0,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,\mathsf{s}(y)) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{Mult}(\mathsf{s}(z),\mathsf{s}(y)) \approx x \\ \qquad \wedge \mathsf{Gt}(\mathsf{s}(y),0) \approx \mathsf{true}, \qquad (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}^{\mathsf{Sim}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \Leftarrow 0 \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,0) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{Mult}(\mathsf{s}(z),0) \approx x \wedge \mathsf{false} \approx \mathsf{true}, \\ \mathsf{Quot}(x,\mathsf{s}(y)) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{s}(\mathsf{Add}(\mathsf{Mult}(z,\mathsf{s}(y)),y)) \approx x \\ \qquad \wedge \mathsf{Gt}(\mathsf{s}(y),0) \approx \mathsf{true}, \qquad (2) \end{array} \right\}$$

$$\equiv \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \Leftarrow 0 \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,\mathsf{s}(y)) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{s}(\mathsf{Add}(\mathsf{Mult}(z,\mathsf{s}(y)),y)) \approx x \\ \quad\quad \wedge \mathsf{Gt}(\mathsf{s}(y),0) \approx \mathsf{true}, \qquad (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Dec}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \Leftarrow 0 \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,\mathsf{s}(y)) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{s}(v) \approx x \wedge v \approx \mathsf{Add}(w,y) \\ \quad\quad \wedge w \approx \mathsf{Mult}(z,\mathsf{s}(y)) \\ \quad\quad \wedge \mathsf{Gt}(\mathsf{s}(y),0) \approx \mathsf{true}, \qquad (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Ded}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(x,y) \approx 0 \Leftarrow 0 \approx x \wedge \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(x,\mathsf{s}(y)) \approx \mathsf{s}(z) \\ \quad \Leftarrow \mathsf{s}(v) \approx x \wedge v \approx \mathsf{Add}(w,y) \\ \quad\quad \wedge \mathsf{Quot}(w,\mathsf{s}(y)) \approx z, \qquad (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Com}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \approx 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \approx \mathsf{s}(\mathsf{Quot}(w,\mathsf{s}(y))) \\ \quad \Leftarrow v \approx \mathsf{Add}(w,y), \qquad (2) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Int}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \approx 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \approx \mathsf{s}(\mathsf{Quot}(w,\mathsf{s}(y))) \\ \quad \Leftarrow v \approx \mathsf{Add}(w,y), \\ (2),(1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Ded}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \approx 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \approx \mathsf{s}(\mathsf{Quot}(w,\mathsf{s}(y))) \\ \quad \Leftarrow w \approx \mathsf{Sub}(v,y), \\ (2),(1) \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Com}}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \approx 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \approx \mathsf{s}(\mathsf{Quot}(\mathsf{Sub}(v,y),\mathsf{s}(y))), \\ (2),(1) \end{array} \right\}$$

$\overset{*}{\Rightarrow}_{\mathcal{R}_{\mathsf{Mult}}}$ (例 4.1 の変換)

$$\overset{*}{\Rightarrow}_{\mathcal{R}_{\mathsf{Mult}}} \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \approx 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \approx \mathsf{s}(\mathsf{Quot}(\mathsf{Sub}(v,y),\mathsf{s}(y))), \\ \mathsf{Sub}(z,0) \approx z, \\ \mathsf{Sub}(\mathsf{s}(w),\mathsf{s}(y)) \approx \mathsf{Sub}(w,y), \\ (2),(1) \end{array} \right\}$$

この変換により Quot を実行可能な CTRS

$$\mathcal{R}_{\mathsf{Quot}} = \left\{ \begin{array}{l} \mathsf{Quot}(0,y) \to 0 \Leftarrow \mathsf{Gt}(y,0) \approx \mathsf{true}, \\ \mathsf{Quot}(\mathsf{s}(v),\mathsf{s}(y)) \to \mathsf{s}(\mathsf{Quot}(\mathsf{Sub}(v,y),\mathsf{s}(y))), \\ \mathsf{Sub}(z,0) \to z, \\ \mathsf{Sub}(\mathsf{s}(w),\mathsf{s}(y)) \to \mathsf{Sub}(w,y) \end{array} \right\}$$

を得る．この $\mathcal{R}_{\mathsf{Quot}}$ は元の仕様 $\mathcal{E}_{\mathsf{Quot}}$ を確かに満たしている．

□

次はもう少し複雑な生成例を紹介する．

［例 4.3］（Subset Sum 問題を解くプログラム SSP の生成例）
SSP は次の入出力関係を持つプログラムである．

- 入力：自然数の多重集合 xs と自然数 v
- 出力：$\sum_{y \in ys} y = v$ を満たす多重集合 $ys \subseteq xs$

多重集合はリストによって表現する．仕様 $\mathcal{E}_{\mathsf{SSP}}$ とライブラリ $\mathcal{R}_{\mathsf{Lib}}$ は以下で定義される．

$$\mathcal{E}_{\mathsf{SSP}} = \left\{ \begin{array}{l} \mathsf{SSP}(xs,v) \approx ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(ys,xs) \approx \mathsf{true} \wedge \mathsf{Sum}(ys) \approx v \end{array} \right\}$$

$$\mathcal{R}_{\mathsf{Lib}} = \left\{ \begin{array}{l} \subseteq_{\mathsf{MS}}(\mathsf{Nil},ys) \to \mathsf{true}, \\ \subseteq_{\mathsf{MS}}(x :: xs,ys) \to \subseteq_{\mathsf{MS}}(xs,\mathsf{Rmv}(x,ys)) \\ \quad \Leftarrow \in_{\mathsf{MS}}(x,ys) \approx \mathsf{true}, \\ \subseteq_{\mathsf{MS}}(x :: xs,ys) \to \mathsf{false} \Leftarrow \in_{\mathsf{MS}}(x,ys) \not\approx \mathsf{true}, \\ \mathsf{Rmv}(x,\mathsf{Nil}) \to \mathsf{Nil}, \\ \mathsf{Rmv}(x,y :: ys) \to ys \Leftarrow x \approx y, \\ \mathsf{Rmv}(x,y :: ys) \to y :: \mathsf{Rmv}(x,ys) \Leftarrow x \not\approx y, \\ \in_{\mathsf{MS}}(x,\mathsf{Nil}) \to \mathsf{false}, \\ \in_{\mathsf{MS}}(x,y :: ys) \to \mathsf{true} \Leftarrow x \approx y, \\ \in_{\mathsf{MS}}(x,y :: ys) \to \in_{\mathsf{MS}}(x,ys) \Leftarrow x \not\approx y, \\ \mathsf{Sum}(\mathsf{Nil}) \to 0, \\ \mathsf{Sum}(x :: xs) \to \mathsf{Add}(\mathsf{Sum}(xs),x), \\ \mathsf{Add}(x,0) \to x, \quad \mathsf{Add}(x,\mathsf{s}(y)) \to \mathsf{s}(\mathsf{Add}(x,y)) \end{array} \right\}$$

$$\left(\begin{array}{l} \mathsf{true} : \mathbb{B}, \quad \mathsf{false} : \mathbb{B}, \quad 0 : \mathbb{N}, \quad \mathsf{s} : \mathbb{N} \to \mathbb{N}, \\ \mathsf{Nil} : \mathbb{N}\ \mathsf{list}, \quad (::) : \mathbb{N} * \mathbb{N}\ \mathsf{list} \to \mathbb{N}\ \mathsf{list}, \\ \mathsf{SSP} : \mathbb{N}\ \mathsf{list} * \mathbb{N} \to \mathbb{N}\ \mathsf{list}, \quad \subseteq_{\mathsf{MS}} : \mathbb{N}\ \mathsf{list} * \mathbb{N}\ \mathsf{list} \to \mathbb{B}, \\ \mathsf{Rmv} : \mathbb{N} * \mathbb{N}\ \mathsf{list} \to \mathbb{N}\ \mathsf{list}, \quad \in_{\mathsf{MS}} : \mathbb{N} * \mathbb{N}\ \mathsf{list} \to \mathbb{B}, \\ \mathsf{Sum} : \mathbb{N}\ \mathsf{list} \to \mathbb{N}, \quad \mathsf{Add} : \mathbb{N} * \mathbb{N} \to \mathbb{N}, \end{array} \right)$$

$\mathcal{R}_{\mathsf{Lib}}$ の下での $\mathcal{E}_{\mathsf{SSP}}$ の変換系列（の 1 つ）を以下に示す．

$\mathcal{E}_{\mathsf{SSP}}$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs,v) \approx \mathsf{Nil} \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(\mathsf{Nil},xs) \approx \mathsf{true} \wedge \mathsf{Sum}(\mathsf{Nil}) \approx v, \\ \mathsf{SSP}(xs,v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(y :: ys,xs) \approx \mathsf{true} \\ \quad\quad \wedge \mathsf{Sum}(y :: ys) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Cas}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs,v) \approx \mathsf{Nil} \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(\mathsf{Nil},xs) \approx \mathsf{true} \wedge \mathsf{Sum}(\mathsf{Nil}) \approx v, \\ \mathsf{SSP}(xs,v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(y :: ys,xs) \approx \mathsf{true} \\ \quad\quad \wedge \in_{\mathsf{MS}}(y,xs) \approx \mathsf{true} \wedge \mathsf{Sum}(y :: ys) \approx v, \\ \mathsf{SSP}(xs,v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(y :: ys,xs) \approx \mathsf{true} \\ \quad\quad \wedge \in_{\mathsf{MS}}(y,xs) \not\approx \mathsf{true} \wedge \mathsf{Sum}(y :: ys) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Sim}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs,v) \approx \mathsf{Nil} \Leftarrow \mathsf{true} \approx \mathsf{true} \wedge 0 \approx v, \\ \mathsf{SSP}(xs,v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(ys,\mathsf{Rmv}(y,xs)) \approx \mathsf{true} \\ \quad\quad \wedge \in_{\mathsf{MS}}(y,xs) \approx \mathsf{true} \\ \quad\quad \wedge \mathsf{Add}(\mathsf{Sum}(ys),y) \approx v, \\ \mathsf{SSP}(xs,v) \approx y :: ys \\ \quad \Leftarrow \mathsf{false} \approx \mathsf{true} \\ \quad\quad \wedge \in_{\mathsf{MS}}(y,xs) \not\approx \mathsf{true} \\ \quad\quad \wedge \mathsf{Add}(\mathsf{Sum}(ys),y) \approx v \end{array} \right\}$$

$$\equiv \left\{ \begin{array}{l} \mathsf{SSP}(xs, v) \approx \mathsf{Nil} \Leftarrow 0 \approx v, \\ \mathsf{SSP}(xs, v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(ys, \mathsf{Rmv}(y, xs)) \approx \mathsf{true} \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(\mathsf{Sum}(ys), y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Dec}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, v) \approx \mathsf{Nil} \Leftarrow 0 \approx v, \\ \mathsf{SSP}(xs, v) \approx y :: ys \\ \quad \Leftarrow \subseteq_{\mathsf{MS}}(ys, zs) \approx \mathsf{true} \wedge \mathsf{Rmv}(y, xs) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \\ \quad\quad \wedge \mathsf{Add}(z, y) \approx v \wedge \mathsf{Sum}(ys) \approx z \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Ded}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, v) \approx \mathsf{Nil} \Leftarrow 0 \approx v, \\ \mathsf{SSP}(xs, v) \approx y :: ys \\ \quad \Leftarrow \mathsf{SSP}(zs, z) \approx ys \wedge \mathsf{Rmv}(y, xs) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Com}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, xs) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(\mathsf{Nil}, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, \mathsf{Nil}) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, \mathsf{Nil}) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, x :: xs) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, x :: xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Cas}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(\mathsf{Nil}, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, \mathsf{Nil}) \approx zs \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, \mathsf{Nil}) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, x :: xs) \approx zs \wedge y \approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, x :: xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Rmv}(y, x :: xs) \approx zs \wedge y \not\approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, x :: xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Sim}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(\mathsf{Nil}, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow \mathsf{Nil} \approx zs \wedge \mathsf{false} \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow xs \approx zs \wedge y \approx x \\ \quad\quad \wedge \mathsf{true} \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow x :: \mathsf{Rmv}(y, xs) \approx zs \wedge y \not\approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\equiv \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow xs \approx zs \wedge y \approx x \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow x :: ws \approx zs \wedge ws \approx \mathsf{Rmv}(y, xs) \wedge y \not\approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Dec}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow xs \approx zs \wedge y \approx x \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow x :: ws \approx zs \wedge ws \approx \mathsf{Rmv}(y, xs) \wedge y \not\approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Int}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow xs \approx zs \wedge y \approx x \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow x :: ws \approx zs \wedge ws \approx \mathsf{Rmv}(y, xs) \wedge y \not\approx x \\ \quad\quad \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \wedge \mathsf{Add}(z, y) \approx v, \\ \mathsf{Sub}(x, y) \approx z \Leftarrow \mathsf{Add}(z, y) \approx x, \\ \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \\ \quad \Leftarrow \mathsf{Rmv}(y, xs) \approx ws \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Ded}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow xs \approx zs \wedge y \approx x \wedge \mathsf{Sub}(v, y) \approx z, \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(zs, z) \\ \quad \Leftarrow x :: ws \approx zs \wedge \mathsf{Sub}(v, y) \approx z \\ \quad\quad \wedge \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \wedge y \not\approx x, \\ \mathsf{Sub}(x, y) \approx z \Leftarrow \mathsf{Add}(z, y) \approx x, \\ \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \\ \quad \Leftarrow \mathsf{Rmv}(y, xs) \approx ws \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \end{array} \right\}$$

$$\overset{*}{\Rightarrow}{}^{\mathsf{Com}}_{\mathcal{R}_{\mathsf{Lib}}} \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \approx \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \approx x :: \mathsf{SSP}(xs, \mathsf{Sub}(v, y)), \\ \mathsf{SSP}(x :: xs, v) \approx y :: \mathsf{SSP}(x :: ws, \mathsf{Sub}(v, y)) \\ \quad \Leftarrow \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \wedge y \not\approx x, \\ \mathsf{Sub}(x, y) \approx z \Leftarrow \mathsf{Add}(z, y) \approx x, \\ \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \\ \quad \Leftarrow \mathsf{Rmv}(y, xs) \approx ws \wedge \in_{\mathsf{MS}}(y, xs) \approx \mathsf{true} \end{array} \right\}$$

紙面の都合上省略するが，この先 Sub, GetItem の生成を行うことにより SSP を実行可能な CTRS

$$\mathcal{R}_{\mathsf{SSP}} = \left\{ \begin{array}{l} \mathsf{SSP}(xs, 0) \to \mathsf{Nil}, \\ \mathsf{SSP}(x :: xs, v) \to x :: \mathsf{SSP}(xs, \mathsf{Sub}(v, y)), \\ \mathsf{SSP}(x :: xs, v) \to y :: \mathsf{SSP}(x :: ws, \mathsf{Sub}(v, y)) \\ \quad \Leftarrow \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, ws) \wedge y \not\approx x, \\ \mathsf{Sub}(z, 0) \to z, \\ \mathsf{Sub}(\mathsf{s}(w), \mathsf{s}(y)) \to \mathsf{Sub}(w, y), \\ \mathsf{GetItem}(x :: xs) \to \mathsf{Tp}(x, xs), \\ \mathsf{GetItem}(x :: xs) \to \mathsf{Tp}(y, x :: zs) \\ \quad \Leftarrow \mathsf{GetItem}(xs) \approx \mathsf{Tp}(y, zs) \wedge x \not\approx y \end{array} \right\}$$

を得る．この $\mathcal{R}_{\mathsf{SSP}}$ は元の仕様 $\mathcal{E}_{\mathsf{SSP}}$ を確かに満たしている．

□

5. 考　　察

5.1 ホーン節の安全な消去

例 4.1 の変換系列を再考する．最初の Expansion において，変換前のホーン節 (1) は変換後の節 2 つによって充足されるので本来は消してもよいが，後の Deduction で必要になるため残しておかなければならない．この節 (1) は最後まで消えずに残っているのだが，元の仕様 $\mathcal{E}_{\mathsf{Sub}}$ は得られた $\mathcal{R}_{\mathsf{Sub}}$ のみで充足されるので最終的に節 (1) は冗長になる．従って，変換の過程で安全に消去できるのが理想である．

5.2 ホーン節中の変数の解釈

次のような例を考える．

[例 5.1]（自然数を 2 倍するプログラム D の生成例）
ライブラリとして，自然数の 1/2 倍を求める CTRS が与えられている．仕様 $\mathcal{E}_{\mathsf{D}}$ とライブラリ $\mathcal{R}_{\mathsf{H}}$ は以下で定義される．

$$\mathcal{E}_{\mathsf{D}} = \left\{ \ \mathsf{D}(x) \approx y \Leftarrow \mathsf{H}(y) \approx x \ \right\}$$

$$\mathcal{R}_{\mathsf{H}} = \left\{ \begin{array}{l} \mathsf{H}(0) \to 0, \\ \mathsf{H}(\mathsf{s}(\mathsf{s}(x))) \to \mathsf{s}(\mathsf{H}(x)) \end{array} \right\}$$

$$\left(\ 0 : \mathbb{N}, \quad \mathsf{s} : \mathbb{N} \to \mathbb{N}, \quad \mathsf{D} : \mathbb{N} \to \mathbb{N}, \quad \mathsf{H} : \mathbb{N} \to \mathbb{N} \ \right)$$

$\mathcal{R}_{\mathsf{H}}$ の下での $\mathcal{E}_{\mathsf{D}}$ の変換系列（の 1 つ）を以下に示す．

$$\begin{array}{l} \mathcal{E}_{\mathsf{D}} \\ \quad \overset{*}{\Rightarrow}{}^{\mathsf{Exp}}_{\mathcal{R}_{\mathsf{H}}} \left\{ \begin{array}{l} \mathsf{D}(x) \approx 0 \Leftarrow \mathsf{H}(0) \approx x, \\ \mathsf{D}(x) \approx \mathsf{s}(y) \Leftarrow \mathsf{H}(\mathsf{s}(y)) \approx x, \\ \mathsf{D}(x) \approx \mathsf{s}(\mathsf{s}(y)) \Leftarrow \mathsf{H}(\mathsf{s}(\mathsf{s}(y))) \approx x \end{array} \right\} \\ \quad \overset{*}{\Rightarrow}_{\mathcal{R}_{\mathsf{H}}} \cdots \overset{*}{\Rightarrow}_{\mathcal{R}_{\mathsf{H}}} \\ \quad \overset{*}{\Rightarrow}{}^{\mathsf{Sim}}_{\mathcal{R}_{\mathsf{H}}} \left\{ \begin{array}{l} \mathsf{D}(0) \approx 0 \Leftarrow 0 \approx x, \\ \mathsf{D}(x) \approx y \Leftarrow \mathsf{H}(\mathsf{s}(y)) \approx x, \\ \mathsf{D}(\mathsf{s}(z)) \approx \mathsf{s}(\mathsf{s}(\mathsf{D}(z))) \end{array} \right\} \end{array}$$

□

この例では，D の関数定義は得られているものの 1 ステップ目に被覆集合 $\{0, \mathsf{s}(y), \mathsf{s}(\mathsf{s}(y))\}$ によって Expansion した際に生じた節の 1 つである，$\mathsf{D}(x) \approx y \Leftarrow \mathsf{H}(\mathsf{s}(y)) \approx x$ が残っている．本稿では，変数の解釈を基底項としているので $\{x \mapsto \mathsf{H}(\mathsf{s}(0)),\ y \mapsto 0\}$ などはこの節の条件部を true にする代入となる．しかし，プログラムにおいて変数には $0, \mathsf{s}(0), \mathsf{s}(\mathsf{s}(0)), \cdots$ の形をした構成子項（データ）が代入されるべきであり，それ以上計算できない項 $\mathsf{H}(\mathsf{s}(0))$ は x の値としては不自然である．

5.3 変換の健全性の証明

定義 3.3 の変換規則がプログラム生成の意味において健全であることは，変換後のホーン節集合が変換前の仕様を満たすことを示すことによって保証される．すなわち，以下の定理が成り立つことを示せばよい．

[定理 5.2]（プログラム生成系の健全性）

$$\mathcal{E} \Rrightarrow_{\mathcal{R}} \mathcal{E}' \quad \text{ならば} \quad \mathcal{E}' \models \mathcal{E} \text{ である．}$$

ただし，変数の解釈は基底項に限定する． □

本稿のプログラム生成系を構築するにあたっては定理 5.2 が成り立つように十分注意したが，実際に証明を書き下してはいない．

6. おわりに

本稿では，プログラム生成系 *GeneSys* の変換規則をホーン節に閉じた変換になるように再構築した．その結果として，変換の過程で論理式の入れ子構造が複雑化することなく直観に近いプログラム生成が可能となることが，いくつかの生成例によって確認できた．

今後の課題としては，前節 5.1 に関連して必要なくなった節は極力消去して不要な変換が適用されないようにすること，5.2 に関連して変数の解釈を構成子項に制限するような述語を加えることでプログラム生成の適用範囲を広げること，5.3 で述べた生成系の健全性の証明を書き下すことがまず挙げられる．その他に自動化への取り組みとして，規則の適用戦略の発見や補題の自動生成は重要な課題である．適切な戦略が確立できない場合でも，近年の潤沢な計算機リソースを利用して brute-force 的な適用法による実装を行うことは，今後の研究にとって有用であろう．

謝辞　本研究は一部，科研費#18500011，#20300010，#20500008，#21700011，及び栢森情報科学振興財団の助成を受けたものである．

文　　献

[1] 植村：“プログラム生成系 GeneSys による融合変換のための戦略”，卒業研究報告，名古屋大学 (2005).
[2] 近藤，酒井，坂部，草刈，西田：“プログラム生成系 GeneSys における等式仕様への否定の導入”，信学技報，SS2007-45 (2007-12), pp. 43–48 (2007).
[3] 長島，酒井，坂部，草刈：“限量子付き等式理論の変換に基づく仕様からのプログラム生成”，コンピュータソフトウェア，**21**, 4, pp. 49–54 (2004).
[4] 西田，酒井，坂部：“構成子項書換え系の逆計算プログラムの生成”，電子情報通信学会論文誌 D-I, **J88-D-I**, 8, pp. 1171–1183 (2005).
[5] M. Ben-Ari: “Mathematical Logic for Computer Science”, 2nd ed., Springer-Verlag, London (2001).
[6] E. Ohlebusch: “Advanced Topics in Term Rewriting”, Springer-Verlag (2002).
[7] D. A. Plaisted: “Semantic confluence tests and completion methods”, Information and Control, **65**, 2/3, pp. 182–215 (1985).
[8] P. L. Wadler: “Deforestation: Transforming programs to eliminate trees”, Theoretical Computer Science, **73**, 2, pp. 231–248 (1990).