マクロ木変換器を融合するための高階木変換器の低階化

佐藤 光　中野 圭介　浅田 和之

マクロ木変換器（MTT）はランク付き木を入出力とするような累積変数付き再帰関数のモデルである．MTT で表現可能な関数のクラスは合成に関して閉じていないため，複数の MTT から等価な一つの MTT を作成する操作（MTT の融合）は必ずしも可能とは限らない．しかし，ある条件の下では MTT の融合が可能であることがいくつかの先行研究から知られている．本研究では，先行研究の中でも特に任意個の MTT の融合が可能な阿部の手法に注目した．この融合手法では複数の MTT から等価な一つの高階木変換器（HTT）を作成し，得られた HTT に関するある条件を満たす場合に限り，その階数を下げる操作（HTT の低階化）を反復することで MTT の融合を達成する．本研究では，同論文で未達成である HTT を低階化するアルゴリズムを提案する．

1 はじめに

マクロ木変換器 (macro tree transducer, MTT) [1] は，ランク付き木を入出力とするような累積変数付き再帰関数のモデルであり，様々な性質が知られている．そのうち，MTT で表現可能な関数のクラスは合成に関して閉じていないことが知られている．よって，複数の MTT と等価な一つの MTT を得る MTT の融合と呼ばれる操作は必ずしも可能とは限らないが，ある条件の下では MTT の融合が可能であることがいくつかの先行研究から知られている．

本研究では，それらの先行研究のうち，下記に示す二つの操作で示されるような融合される MTT の個数に関する制約を要求しない手法 [5] に注目しており，同論文で未達成である高階木変換器の低階化を行うための条件とその手法の定式化を検討する．

1. (**HTT の作成**) n 個の MTT に対して，等価な n- 高階木変換器 (high-level tree transducer, HTT) [5] を作成する．
2. (**HTT の低階化**) 作成された n-HTT に対して，等価な $(n-1)$-HTT を一定の条件下において作成する操作を反復する．

Level reduction of high-level tree transducer to compose macro tree transducer.

Akira Sato, Keisuke Nakano, Kazuyuki Asada,, 東北大学電気通信研究所, Research Institute of Electrical Communication, Tohoku University.

2 準備

本節ではランク付きアルファベット，木，型および累積変数の定義を示す．はじめに，見やすさのための略記等の記法を定義する．

定義 2.1 (記法) 本論文では自然数 $\mathbb{N}$ を 0 以上の整数としている．加えて，以下の記法を用いる．

- $n \in \mathbb{N} \setminus \{0\}$ に対して $[n]$ を以下で定義する．
$$[n] ::= \{i \mid 1 \leq i \leq n\}$$
- $n \in \mathbb{N}$ に対して $[n]_0$ を以下で定義する．
$$[n]_0 ::= \{i \mid 0 \leq i \leq n\}$$
- k を $\mathbb{N}$ から $\mathbb{N}$ への関数とする．このとき $i \in \mathbb{N}$ に対して $k(i)$ を k_i と表記する．
- $r \in \mathbb{N}$ および記号 x に対して組 (x^r) を以下で定義する．
$$(x^r) ::= (\overbrace{x, x, \ldots, x}^{r\ 個})$$
- $i \in \mathbb{N}$ および記号 y に対して変数の組 (Y_i) およびそのサイズ $|Y_i|$ を以下で定義する．
$$(Y_i) ::= (y_{i,1}, y_{i,2}, \ldots, y_{i,k_i})$$
$$|Y_i| ::= k_i$$

続いて，ランク付きアルファベットおよび木を定義

する．

定義 2.2 (ランク付きアルファベット) **ランク付きアルファベット**は二項組 $(\Sigma, \mathrm{rank}_\Sigma)$ である．ここで Σ は記号の有限集合，rank_Σ は Σ から $\mathbb{N}$ への関数である．$\sigma \in \Sigma$ に対して $\mathrm{rank}_\Sigma(\sigma)$ を σ **のランク**と呼ぶ．Σ の要素のうち，ランク n の全ての要素からなる集合を $\Sigma^{(n)}$ と表記する．以降はこのランク付きアルファベットを単に Σ と表記する．

定義 2.3 (木) ランク付きアルファベット Σ に対する**木**全体の集合を T_Σ と表記し，下記を満たす最小の集合 T として定義する．

$$\sigma \in T \quad (\sigma \in \Sigma^{(0)})$$
$$\sigma(t_1, t_2, \ldots, t_r) \in T \quad (\sigma \in \Sigma^{(r)}, t_1, t_2, \cdots, t_r \in T)$$

ランク付きアルファベット Δ に対して，その木全体の集合 T_Δ と対応する基底型を o_Δ と表す．MTT および HTT の累積変数，非終端記号および状態の定義を行うために型および状態型を定義する．

定義 2.4 (型) ランク付きアルファベット Δ に対して，Δ **上の型** τ を以下で定義する．

$$\tau ::= \tau^{(0)} \mid \tau^{(1)} \mid \cdots$$
$$\tau^{(0)} ::= \mathrm{o}_\Delta$$
$$\tau^{(i+1)} ::= (\tau^{(i)}, \ldots, \tau^{(i)}) \to \tau^{(i)}$$

Δ 上の型 τ 全体の集合を $\mathit{Type}(\Delta)$ と表す．**型** τ **のレベル** $lv(\tau)$ を $lv(\tau^{(i)}) = i$ と定める．

定義 2.5 (状態型) ランク付きアルファベット Σ，Δ に対して，Σ，Δ **上の状態型** τ_q を以下で定義する．

$$\tau_q ::= \mathrm{o}_\Sigma \to \tau \qquad (\tau \in \mathit{Type}(\Delta))$$

Σ，Δ 上の状態型 τ_q 全体の集合を $\mathit{QType}(\Sigma, \Delta)$ と表す．**状態型** τ_q **のレベル** $qlv(\tau_q)$ を $qlv(\mathrm{o}_\Sigma \to \tau) = lv(\tau)$ と定める．

型を持つ非終端記号および状態型を持つ状態の定義を行うために型付きアルファベットおよび状態型付きアルファベットを定義する．

定義 2.6 (型付きアルファベット) Δ **上の型付きアルファベット**は (M, type_M) である．ここで M は記号の有限集合，type_M は M から $\mathit{Type}(\Delta)$ への関数である．$m \in M$ に対して $\mathrm{type}_M(m)$ を m の**型**と呼ぶ．型が τ の要素全てからなる集合 $\{m \in M \mid \mathrm{type}_M(m) = \tau\}$ を $M^{(\tau)}$ と表記する．以降はこの型付きアルファベットを単に M と表記する．

定義 2.7 (状態型付きアルファベット) Σ, Δ **上の状態型付きアルファベット**は (Q, qtype_Q) である．ここで Q は**状態**の有限集合，qtype_Q は Q から $\mathit{QType}(\Sigma, \Delta)$ への関数である．$q \in Q$ に対して $\mathrm{qtype}_Q(q)$ を q の**状態型**と呼ぶ．以降はこの状態型付きアルファベットを単に Q と表記する．

型を持つ累積変数の定義を行うために累積変数用変数集合と呼ばれる型付きアルファベットを定義する．

定義 2.8 (累積変数用変数集合) レベルが m であるような Δ 上の型

$$\tau = (\tau_{m-1,1}, \ldots, \tau_{m-1,k_{m-1}}) \to \cdots \to (\tau_{0,1}, \ldots, \tau_{0,k_0}) \to \mathrm{o}_\Delta$$

に対して，**累積変数用変数集合**を Δ 上の型付きアルファベット $\mathcal{Y}_\tau$ として定義する．

$$\mathcal{Y}_\tau ::= \{y_{i,j} \mid i \in [m-1]_0, j \in [k_i]\}$$
$$\mathrm{type}_{\mathcal{Y}_\tau}(y_{i,j}) ::= \tau_{i,j} \qquad (lv(\tau_{i,j}) = i)$$

3 高階木変換器

本節では高階木変換器，マクロ木変換器およびマクロ木変換器の融合を定義する．はじめに高階木変換器を定義する．

定義 3.1 (高階木変換器 (HTT)) **高階木変換器 (HTT)** $\mathcal{H}$ は七項組 $(Q, M, \Sigma, \Delta, \mathit{init}, R, P)$ である．

- Σ は**入力アルファベット**と呼ばれるランク付きアルファベットである．ただし $\Sigma^{(0)} \neq \emptyset$ である．
- Δ は**出力アルファベット**と呼ばれるランク付きアルファベットである．ただし $\Delta^{(0)} \neq \emptyset$ である．
- Q は Σ，Δ 上の状態型付きアルファベットであり，この集合の要素は**状態**と呼ばれる．
- M は Δ 上の型付きアルファベットであり，この集合の要素は**非終端記号**と呼ばれる．
- $\mathit{init} \in \mathcal{T}_{Q,M,\Delta}(\{x_1\}, \emptyset)^{(\mathrm{o}_\Delta)}$ は**初期表現**と呼ばれる HTT 項である．
- R は**規則集合**と呼ばれる各 $q \in Q^{(\mathrm{o}_\Sigma \to \tau)}$，$\sigma \in \Sigma^{(s)}$ に対して丁度一つの (q, σ)**-規則** $\mathit{lhs}(q, \sigma) \to \mathit{rhs}(q, \sigma)$ を含む有限集合である．ただし $\mathit{lhs}(q, \sigma)$ および $\mathit{rhs}(q, \sigma)$ は入力木の部分木 $x_1, \ldots, x_s \in T_\Sigma$ および累積変数用変数集合

$$\frac{\mathrm{type}_F(f) = \mathrm{o}_\Sigma \to \tau \qquad x \in \mathcal{X}}{f\langle x\rangle : \tau}\ (\mathsf{fc})$$

$$\frac{\mathrm{type}_M(m) = \tau}{m : \tau}\ (\mathsf{nt})$$

$$\frac{\mathrm{rank}_\Delta(\delta) = s \qquad \tau = (\mathrm{o}_\Delta^s) \to \mathrm{o}_\Delta}{\delta : \tau}\ (\mathsf{sym})$$

$$\frac{\mathrm{type}_\mathcal{Y}(y) = \tau \qquad y \in \mathcal{Y}}{y : \tau}\ (\mathsf{var})$$

$$\frac{H_0 : (\tau_1, \ldots, \tau_s) \to \tau \qquad \forall i \in [s].H_i : \tau_i}{H_0(H_1, \ldots, H_s) : \tau}\ (\mathsf{app})$$

図 1　HTT 項の型付け規則

$\mathcal{Y}_\tau$ を用いて，次を満たすものとする．

$$lhs(q,\sigma) = q\langle\sigma(x_1,\ldots,x_s)\rangle(Y_{m-1})\cdots(Y_0)$$
$$rhs(q,\sigma) \in \mathcal{T}_{Q,M,\Delta}(\{x_1,\ldots,x_s\},\mathcal{Y}_\tau)$$

- P は**文法集合**と呼ばれる各 $m \in M^{(\tau)}$ に対して丁度一つの (m)**-文法** $lhs(m) \to rhs(m)$ を含む集合である．ただし $lhs(m)$ および $rhs(m)$ は $M' = \{m' \in M \mid lv(m') < lv(\tau)\}$ および累積変数用変数集合 $\mathcal{Y}_\tau$ を用いて，次を満たすものとする．

$$lhs(m) = m(Y_{m-1})\cdots(Y_0)$$
$$rhs(m) \in \mathcal{T}_{Q,M',\Delta}(\emptyset,\mathcal{Y}_\tau)$$

ここで $\mathcal{X} \subseteq T_\Sigma$ および Δ 上の型付きアルファベット $\mathcal{Y}$ に対して集合 $\mathcal{T}_{Q,M,\Delta}(\mathcal{X},\mathcal{Y})$ は型付け可能な **HTT 項**全体の集合である．HTT 項 H を以下で定義する．

$$\begin{aligned} M ::= {} & q\langle x\rangle && (q \in Q, x \in \mathcal{X}) \\ \mid {} & m && (m \in M) \\ \mid {} & \delta && (\delta \in \Delta) \\ \mid {} & y && (y \in \mathcal{Y}) \\ \mid {} & M(M,\ldots,M) \end{aligned}$$

HTT 項 H が**型付け可能**とは，図 1 が示す型付け規則により判断 $H : \tau$ が導出可能であることを意味する．このとき τ を H の**型**と呼ぶ．また，H の型は一意に定まるため，これを $\mathrm{type}(H)$ で表す．任意の状態 $q \in Q$ および非終端記号 $m \in M$ のレベルが n 以下のとき，$\mathcal{H}$ を n-HTT と呼ぶ．

定義 3.2 (HTT に関する導出関係) HTT 項 H 上の**導出関係** $\Rightarrow_\mathcal{H}$ を次で構成する．

- 任意の状態 $q \in Q^{(\mathrm{o}_\Sigma \to \tau)}$，記号 $\sigma \in \Sigma^{(r)}$ および $s ::= lv(\tau)$ とすると $\forall i \in [s-1]_0, j \in [k_i], h_{i,j} \in H_i.\ lv(\mathrm{type}(h_{i,j})) = i$ であるような HTT 項の列 $H_{s-1},\ldots,H_0$ に対して，入力木の部分木 $x_1,\ldots,x_r \in T_\Sigma$ および累積変数用変数集合 $\mathcal{Y}_\tau$ を用いて

$$q\langle\sigma(x_1,\ldots,x_r)\rangle(Y_{s-1})\cdots(Y_0) \to rhs(q,\sigma) \in R$$

であるとき，次が成り立つ．

$$q\langle\sigma(t_1,\ldots,t_r)\rangle(H_{s-1})\cdots(H_0) \Rightarrow_\mathcal{H} rhs(q,\sigma)[x_i \leftarrow t_i]_{i\in[r]}[y_{i,j} \leftarrow h_{i,j}]_{i\in[s-1]_0, j\in[k_i]}$$

- 任意の非終端記号 $m \in M^{(\tau)}$ および $s ::= lv(\tau)$ とすると $\forall i \in [s-1]_0, j \in [k_i], h_{i,j} \in H_i.\ lv(\mathrm{type}(h_{i,j})) = i$ であるような HTT 項の列 $H_{s-1},\ldots,H_0$ に対して，累積変数用変数集合 $\mathcal{Y}_\tau$ を用いて

$$m(Y_{s-1})\cdots(Y_0) \to rhs(m) \in P$$

であるとき，次が成り立つ．

$$m(H_{s-1})\cdots(H_0) \Rightarrow_\mathcal{H} rhs(m)[y_{i,j} \leftarrow h_{i,j}]_{i\in[s-1]_0, j\in[k_i]}$$

- 任意の $\delta \in \Delta^{(r)}$，HTT 項 $H_1,\ldots,H_r$，$i \in [r]$ に対して $H_i \Rightarrow_\mathcal{H} H_i'$ であるとき，次が成り立つ．

$$\delta(H_1,\ldots,H_i,\ldots,H_r) \Rightarrow_\mathcal{H} \delta(H_1,\ldots,H_i',\ldots,H_r)$$

補題 3.3 (導出 $\Rightarrow_\mathcal{H}$ の正規形の存在と唯一性) 導出関係 $\Rightarrow_\mathcal{H}$ に対して任意の HTT 項 $H \in \mathcal{T}_\mathcal{H}^{(\mathrm{o}_\Delta)}$ は唯一の正規形 $H' \in T_\Delta$ を持つ．

定義 3.4 (HTT の木変換) 導出 $\Rightarrow_\mathcal{H}$ に関する HTT 項 $H^{(\mathrm{o}_\Delta)}$ の正規形を $[\![H]\!]_\mathcal{H}$ とし，これを H の**評価**と呼ぶ．HTT **の木変換** $Tr(\mathcal{H}) : T_\Sigma \to T_\Delta$ を $Tr(\mathcal{H})(t) = [\![init[x_1 \leftarrow t]]\!]_\mathcal{H}$ と定義する．

続いて，マクロ木変換器の定義を示す．

定義 3.5 (マクロ木変換器 (MTT)) **マクロ木変換器 (MTT)** は非終端記号を含まない 1-HTT である．例として，入力木 t に対して反転した木を計算する MTT $\mathcal{M}_{reverse}$ を以下に示す．

例 3.6 $\mathcal{M}_{reverse} = (Q, M, \Sigma, \Sigma, init, R, P)$ を以下に

定義する．

$$
\begin{aligned}
Q &= \{q\}, M = \emptyset, \\
\Sigma^{(0)} &= \{e\},\ \Sigma^{(1)} = \{a, b\},\ \forall n > 1. \Sigma^{(n)} = \emptyset, \\
init &= q\langle x_1\rangle(e), \\
R &= \{q\langle a(t_1)\rangle(y_1) \to q\langle t_1\rangle(a(y_1)), \\
&\quad q\langle b(t_1)\rangle(y_1) \to q\langle t_1\rangle(b(y_1)), \\
&\quad q\langle e\rangle(y_1) \to y_1\}, P = \emptyset.
\end{aligned}
$$

続いて，MTT の融合の定義を示す．

定義 3.7 (MTT の融合) n 個の MTT$\mathcal{M}_1, \ldots, \mathcal{M}_n$ に対して，下記を満たすような MTT$\mathcal{M}^*$ を得る操作を **MTT の融合**と定義する．

$$\forall t. Tr(\mathcal{M}^*)(t) = Tr(\mathcal{M}_n) \circ \cdots \circ Tr(\mathcal{M}_1)(t)$$

4 HTT の低階化手法

本節では **(1) 新たな状態・非終端記号の作成**，**(2) 新たな規則・文法の作成**および **(3) 既存の規則・文法・初期表現の書き換え**の三つの操作によって達成される HTT の低階化について述べる．

4.1 新たな状態・非終端記号とその規則・文法の作成

$q \in Q^{(o \to \tau)}$（ただし $lv(\tau) = n \geq 2$）および $\sigma \in \Sigma^{(r)}$ に対して，累積変数用変数集合 $\mathcal{Y}_\tau$ を用いて以下のように規則の左辺 $lhs(q, \sigma)$ を表現することができる．

$$lhs(q, \sigma) ::= q\langle\sigma(x_1, \ldots, x_r)\rangle(Y_{n-1}) \cdots (Y_0) \quad (1)$$

このとき，レベル $n-1$ の累積変数である $y_{n-1,j} \in Y_{n-1}$ が規則の右辺 $rhs(q, \sigma)$ に出現する場合，定義より次の四通りに場合分けできる．ただしここでは $(i, j) \in [n-1]_0 \times [k_{y,i}]$，$r' \in [r]$ である．

$$rhs(q, \sigma) ::= \begin{cases} q'\langle x_{r'}\rangle(Z_{i-1}) \cdots (Z_0) & \text{(q-st)} \\ m(Z_{i-1}) \cdots (Z_0) & \text{(q-nt)} \\ \delta(H_1, \ldots, H_s) & \text{(q-sym)} \\ y_{i,j}(Z_{i-1}) \cdots (Z_0) & \text{(q-app)} \end{cases}$$

同様に，各 $m \in M^{(\tau)}$ に対して，累積変数用変数集合 $\mathcal{Y}_\tau$ を用いて以下のように規則の左辺 $lhs(m)$ を表現することができる．

$$lhs(m) ::= m(Y_{n-1}) \cdots (Y_0) \quad (2)$$

このとき，レベル $n-1$ の累積変数である $y_{n-1,j} \in Y_{n-1}$ が規則の右辺 $rhs(q, \sigma)$ に出現する場合，定義より次の二通りに場合分けできる．

$$rhs(m) ::= \begin{cases} \delta(H_1, \ldots, H_s) & \text{(m-sym)} \\ y_{i,j}(Z_{i-1}) \cdots (Z_0) & \text{(m-app)} \end{cases}$$

ここでレベル $n-1$ の累積変数 $y_{n-1,j}$ の右辺 (すなわち $rhs(q, \sigma)$ または $rhs(m)$) に出現する場合，以下の三つの制約を満たす場合に限り，低階化が可能である．

1. (**複製回数制約**) 一つの変数ごとに高々一つ出現する．例えば以下のようなレベル 2 の状態に対応する規則に対して，レベル 1 の状態を用いた規則への書き換え操作は不可能である．

   $$q\langle e\rangle(y_{1,1})(y_{0,1}) \to y_{1,1}(y_{1,1}(y_{0,1}))$$

   これは $y_{1,1}$ が将来的に受け取る HTT 項を計算するために新たな状態と規則を追加する際，右辺に $y_{1,1}$ が複数個出現すると将来的に受け取る HTT 項が一つに定まらないためである．
2. (**出現形式制約**) 関数として，あるいは状態か非終端記号の引数として出現する．すなわち $y_{n-1,j}$ が (q-app) および (m-app) における $y_{i,j}$ に相当するか，あるいは (q-st) および (q-nt) の $Z_{i-1}, \ldots, Z_0$ の部分で出現する．もちろん (q-sym) や (m-sym) の $H_1, \ldots, H_s$ の部分に出現する場合も含まれる．
3. (**引数階数制約**) 状態か非終端記号の引数として出現する場合，レベル $n-1$ 累積変数の組の要素内に出現する．すなわち (q-st) および (q-nt) の $Z_{i-1}, \ldots, Z_0$ の部分で出現する場合，特に $i = n$ かつ Z_{n-1} の部分に出現していなければならない．例えば，下に示すような規則はこの制約を満たしていない．

   $$\begin{aligned} &q\langle\sigma(x_1, x_2)\rangle(y_{2,1}, y_{2,2})(y_{1,1})(y_{0,1}) \\ &\qquad \to q'\langle x_1\rangle(y_{2,1})(y_{2,2}(y_{1,1}))(y_{0,1}) \end{aligned}$$

   この例の場合はレベル 2 の HTT 項（右辺の $y_{2,1}$）が将来的に受け取る HTT 項は新しい規則により計算できる．一方，レベル 1 以下の HTT 項（右辺の $y_{2,2}(y_{1,1})$）が将来どのような HTT 項を受け取るのかを計算することは出来ない．左辺では $y_{2,2}$ はレベル 2 の変数の組 $(y_{2,1}, y_{2,2})$ に位置するが，右辺ではレベル 1 の変数の組 $(y_{2,2}(y_{1,1}))$ に位置しているため，将来的に受け取る HTT 項を計算するためにはレベル 1 の累積変数が将来

的に受け取る HTT 項を計算するための状態および規則を追加しなければならない．

これら三制約を満たす場合，以下の手順で状態 (非終端記号) と，それに対応する規則 (文法) を作成する．

1. (**状態・非終端記号の作成**)
   状態 $q \in Q$ に対して (q,σ)-規則の左辺 $lhs(q,\sigma)$(式 1) におけるレベル $n-1$ 累積変数の組 Y_{n-1} が $Y_{n-1} = y_{n-1,1},\ldots,y_{n-1,k_{y,n-1}}$ であるとする．このとき $y_{n-1,j}(j \in [k_{y,n-1}])$ に対して式 3 が成り立つとすると，新たに状態 $q_{j,(i',j')}(i' \in [n-2]_0, j' \in [k_{y',i'}])$ を作成する．
   また，非終端記号 $m \in M$ に対しては m-文法の左辺 $lhs(m)$(式 2) におけるレベル $n-1$ 累積変数の組 Y_{n-1} が $Y_{n-1} = y_{n-1,1},\ldots,y_{n-1,k_{y,n-1}}$ である場合とする．このとき $y_{n-1,j}(j \in [k_{y,n-1}])$ に対して式 3 が成り立つとすると，新たに状態 $m_{j,(i',j')}(i' \in [n-2]_0, j' \in [k_{y',i'}])$ を作成する．
   $$\begin{aligned}&\mathrm{type}_{\mathcal{Y}_\tau}(y_{n-1,j})\\&\quad=(y_{n-2,1},\ldots,y_{n-2,k_{y',n-2}})\\&\qquad\cdots(y_{0,1},\ldots,y_{0,k_{y',0}})\end{aligned}\tag{3}$$
2. (**規則 (文法) の左辺の作成**)
   (q,σ)-規則の左辺 $lhs(q,\sigma)$(式 1) におけるレベル $n-2$ 以下の累積変数の部分 $(Y_{n-2})\cdots(Y_0)$ を用いて，レベル $n-1$ の累積変数 $y_{n-1,j}(j \in [k_{y,n-1}])$ に対して $(q_{j,(i',j')},\sigma)$-規則の左辺 $lhs(q_{j,(i',j')},\sigma)$ を以下で定める．
   $$\begin{aligned}&lhs(q_{i,(i',j')},\sigma)\\&\qquad ::= q_{i,(i',j')}\langle\sigma(\ldots)\rangle(Y_{n-2})\cdots(Y_0)\end{aligned}$$
   (m)-規則の左辺 $lhs(m)$(式 2) におけるレベル $n-2$ 以下の累積変数の部分 $(Y_{n-2})\cdots(Y_0)$ を用いて，レベル $n-1$ の累積変数 $y_{n-1,j}(j \in [k_{y,n-1}])$ に対して $m_{j,(i',j')}$-文法の左辺 $lhs(m_{j,(i',j')})$ を以下で定める．
   $$lhs(m_{i,(i',j')}) ::= m_{j,(i',j')}(Y_{n-2})\cdots(Y_0)$$
3. (**規則 (文法) の右辺の作成**)
   レベル $n-1$ 累積変数 $y_{n-1,j}(j \in [k_{y,n-1}])$ に対して (q,σ)-規則または m-文法の右辺（これらの右辺をまとめて rhs と表す）における累積変数 $y_{n-1,j}$ の出現パターンで場合分けし，さらに一部の場合においては再帰的な計算を行うようなアルゴリズム CreateRules(rhs,j) を適用する．これにより，累積変数 $y_{n-1,j}$ が将来的に引数として受け取る HTT 項 $z_{i',j'}$ に対応する $(q_{i,(i',j')},\sigma)$-規則（または $m_{i,(i',j')}$-文法）の右辺を定める．

ここで，CreateRules(rhs,j) は以下のようなアルゴリズムである．ここで $rhs ::= rhs(q,\sigma) \mid rhs(m)$, $rhs_{\mathrm{id}} ::= rhs(q_{\mathrm{id}},\sigma) \mid rhs(m_{\mathrm{id}})$ を定義する．

1. $rhs = y_{n-1,j}(Z_{n-2})\cdots(Z_0)$ の場合
   $$rhs_{j,(i',j')} ::= z_{i',j'}(Y_{n-i'-1})\cdots(Y_0)$$
2. $rhs = q'\langle x\rangle(Z_{n-1})\cdots(Z_0)(q' \in Q)$ の場合
   $y_{1,j} \lhd z_{n-1,h} \in Z_{n-1}$ であるとする．
   各 $i' = 0,\ldots,n-2$, $j' = 1,\ldots,|Z_{i'}|$ に対して $w_{i',j'}$ および $W_{i'}$ を以下で定義する．
   $$\begin{aligned}w_{i',j'} &::= q'_{h,(i',j')}\langle x\rangle(Z_{n-2})\cdots(Z_{i'})\\W_{i'} &::= w_{i',1},\ldots,w_{i',|Z_{i'}|}\end{aligned}$$
   続いて，以下のように再帰呼び出しを行う．
   CreateRules$(z_{n-2,h}(W_{n-2})\cdots(W_0),j)$
3. $rhs = m'(Z_{n-1})\cdots(Z_0)(m' \in M)$ の場合
   $y_{1,j} \lhd z_{n-1,h} \in Z_{n-1}$ であるとする．
   各 $i' = 0,\ldots,n-2$, $j' = 1,\ldots,|Z_{i'}|$ に対して $w_{i',j'}$ および $W_{i'}$ を以下で定義する．
   $$\begin{aligned}w_{i',j'} &::= m'_{h,(i',j')}(Z_{n-2})\cdots(Z_{i'})\\W_{i'} &::= w_{i',1},\ldots,w_{i',|Z_{i'}|}\end{aligned}$$
   続いて，以下のように再帰呼び出しを行う．
   CreateRules$(z_{n-2,h}(W_{n-2})\cdots(W_0),j)$

以上の操作により規則 (文法) が作られる．

4.2 規則 (文法，初期表現) の書き換え

規則 (文法) の書き換えは二段階に分けられる．一つが規則 (文法) の左辺の書き換えであり，もう一つが規則 (文法) の右辺の書き換えである．また，初期表現は規則 (文法) の右辺の書き換えと同じ手法で書き換えることができる．

1. (**規則 (文法) 左辺 lhs の書き換え**)
   $q \in Q$, $m \in M$ に対して $F ::= q\langle x\rangle \mid m$ を定義する．このとき規則 (文法) の左辺 lhs は $F(Y_{n-1})(Y_{n-2})\cdots(Y_0)$ と表される．レベル $n-1$ 累積変数の組 Y_{n-1} と，その要素であるレベル $n-1$ 累積変数 $y_{n-1,j}$ に対して，$Y'_{n-1} ::= y'_{n-1,1},\ldots,y'_{n-1,k_{y,n-1}}$ を定義する．

また，$y_{n-1,j}$ が将来的にレベル $n-2$ 累積変数を受け取った結果の HTT 項を $y'_{n-1,j}$ と定義しており，式 3 から以下を満たす．

$$\begin{aligned}&\mathrm{type}_{Y'_{n-1}}(y'_{n-1,j})\\&\quad=(y_{n-3,1},\ldots,y_{n-3,k_{y',n-3}})\\&\qquad\qquad\cdots(y_{0,1},\ldots,y_{0,k_{y',0}})\end{aligned}$$

ここで書き換えられた規則または文法の左辺 lhs' を以下に示す．

$$lhs' ::= F(Y_{n-2},Y'_{n-1})(Y_{n-3})\cdots(Y_0)$$

2. **(規則 (文法) 右辺 rhs および初期表現 $init$ の書き換え)**

$q\in Q$，$\sigma\in\Sigma$，$m\in M$ に対して $rhs ::= rhs(q,\sigma)\mid rhs(m)\mid init$ とする．このとき，以下のアルゴリズム Rewrite(rhs) により rhs' を構成する．

(a) $rhs=\delta(H_1,\ldots,H_s)$ の場合

$rhs ::= \delta(\mathrm{Rewrite}(H_1),\ldots,\mathrm{Rewrite}(H_s))$

(b) $rhs=y_{n-1,j}(Z_{n-2})\cdots(Z_0)$ の場合

$rhs' ::= y'_{n-1,j}(Z_{n-3})\cdots(Z_0)$

(c) $rhs=q'\langle x\rangle(Z_{n-1})\cdots(Z_0)(q'\in Q)$ の場合

$rhs' ::= \mathrm{RewriteRules}(Y_{n-1},rhs(q,\sigma))$

(d) $rhs=m'(Z_{n-1})\cdots(Z_0)(m'\in M)$ の場合

$rhs' ::= \mathrm{RewriteGrammars}(Y_{n-1},rhs(m))$

最後に，RewriteRules($Z_{n-1},rhs(q,\sigma)$) および RewriteGrammars($Z_{n-1},rhs(m)$) を図 2 と図 3 に示す．

5 マクロ木変換器の融合に関する過去の研究

本節では本論文に関係する過去の研究について述べ，それらと本論文が採用した融合手法との比較を行う．はじめに，三つの過去の研究について述べる．

- Maneth [3] の任意個の MTT 融合可能性に関する研究

  Maneth は木変換の合成が入出力木のサイズに関する一定の制約を満たす木変換のクラスに含まれている場合に融合が可能であることを示した．この融合手法は MTT と異なる複数の計算モデルを経由しており，直接的に MTT を構成するわ

INPUT: Z_{n-1}，$q'\langle x'\rangle(W_{n-1})\cdots(W_0)$
for all $(h,j)\in[k_{w,n-1}]\times[k_{z,n-1}]$
  if $w_{n-1,h}=z_{n-1,j}$ then
    $w'_{n-1,h} ::= z'_{n-1,j}$
  if $w_{n-1,h}=\delta(H_1,\ldots,H_s)$ then
    $w'_{n-1,h}$
    $::= \delta(\mathrm{RewriteRules}(Z_{n-1},H_1),$
      $\ldots,\mathrm{RewriteRules}(Z_{n-1},H_s))$
  else if $w_{n-1,h}=q''\langle x''\rangle(U_{n-1})$ then
    for all $i'\in[n-2]_0$
      for all $j'\in[k_{u,i'}]$
        $u_{i',j'} ::= q'_{h,(i',j')}\langle x'\rangle(W_{n-2})\cdots(W_0)$
      $U_{i'} ::= u_{i',1},\ldots,u_{i',k_{u,i'}}$
    $sub ::= q''\langle x''\rangle(U_{n-1})\cdots(U_0)$
    $w'_{n-1,h} ::= \mathrm{RewriteRules}(Z_{n-1},sub)$
  else if $w_{n-1,h}=m''(U_{n-1})$ then
    for all $i'\in[n-2]_0$
      for all $j'\in[k_{u,i'}]$
        $u_{i',j'} ::= q'_{h,(i',j')}\langle x'\rangle(W_{n-2})\cdots(W_0)$
      $U_{i'} ::= u_{i',1},\ldots,u_{i',k_{u,i'}}$
    $sub ::= m''(U_{n-1})\cdots(U_0)$
    $w'_{n-1,h} ::= \mathrm{RewriteGrammars}(Z_{n-1},sub)$
OUTPUT: $q'\langle x'\rangle(W_{n-2},W'_{n-1})\cdots(W_0)$

図 2 RewriteRules($Z_{n-1},rhs(q,\sigma)$)

INPUT: Z_{n-1}，$m'(W_{n-1})\cdots(W_0)$
for all $(h,j)\in[k_{w,n-1}]\times[k_{z,n-1}]$
  if $w_{n-1,h}=z_{n-1,j}$ then
    $w'_{n-1,h} ::= z'_{n-1,j}$
  if $w_{n-1,h}=\delta(H_1,\ldots,H_s)$ then
    $w'_{n-1,h}$
    $::= \delta(\mathrm{RewriteRules}(Z_{n-1},H_1),$
      $\ldots,\mathrm{RewriteRules}(Z_{n-1},H_s))'$
OUTPUT: $m'(W_{n-2},W'_{n-1})\cdots(W_0)$

図 3 RewriteGrammars($Z_{n-1},rhs(m)$)

けではない.

- Voigtländer と Kühnemann [4] による二つの MTT に対する融合手法の提案
  Voigtländer と Kühnemann は累積変数の使用に関する一定の制約の下で二つの MTT を融合する手法を提案した．しかし，融合後の MTT が累積変数の使用に関する制約を再度満たすとは限らないため，三つ以上の MTT の融合には適さない.
- Engelfriet [2] による任意個の MTT と HTT の変換に関する研究
  Engelfriet は n 個の MTT と n-HTT の表現可能なクラスが一致することを示した．構成的な証明方法であるが，Maneth 同様に別の計算モデルを経由している.

阿部の論文では任意個の MTT に対する実用的な融合手法として，複数の MTT の合成と等価な HTT の構成方法および HTT の低階化の一例が示されていた．本論文では，阿部の示した例に基づいた HTT の低階化手法を提案し，さらにこの手法が適用できる HTT の規則 (文法) が満たすべき条件に関して述べた.

6 まとめと今後の課題

本論文では複数の MTT を融合する手法を実現する目的で，融合の過程で作成される HTT に対する低階化手法を提案した．今後の研究では，この低階化手法の正当性を示し，さらに融合前後での計算量の変化などを考察する.

参 考 文 献

[1] Engelfriet, J. and Vogler, H.: Macro tree transducers, *Journal of Computer and System Sciences*, Vol. 31, No. 1(1985), pp. 71–146.

[2] Engelfriet, J. and Vogler, H.: High level tree transducers and iterated pushdown tree transducers, *Acta Informatica*, Vol. 26, No. 1-2(1988), pp. 131–192.

[3] Maneth, S.: The Macro Tree Transducer Hierarchy Collapses for Functions of Linear Size Increase, *Foundations of Software Technology and Theoretical Computer Science (FST TCS)*, Vol. LNCS 2914(2003), pp. 326–337.

[4] Voigtländer, J. and Kühnemann, A.: Composition of functions with accumulating parameters, *Journal of Functional Programming*, Vol. 14, No. 3(2004), pp. 317–363.

[5] 阿部和敬: 高階木変換器を経由したマクロ木変換器の直接的融合 電気通信大学大学院情報理工学研究科修士論文（未公刊）, (2019).