# 修 士 論 文 の 和 文 要 旨

| | | | |
|---|---|---|---|
| 研究科・専攻 | 大学院　情報理工　学研究科　情報・ネットワーク工学専攻　博士前期課程 | | |
| 氏　　名 | 高橋 祐多 | 学籍番号 | 1731095 |
| 論 文 題 目 | ランク付き木から文字列への決定性ストリーム変換器の表現力 | | |


要　旨

構造化されたデータに対する変換は，関数プログラミングや文書処理において重要な役割を果しており，そのような変換のふるまいのよいクラスを明らかにするため，様々な計算モデルが提案されている．そのひとつとして，Alur と D'Antoni らによって提案された単 一使用制約付き決定性ストリーム木変換器 ($STT_{sur}$) がある．$STT_{sur}$ はネスト文字列上の変換を定義することができ，ネスト文字列はランク付き木や森といった階層構造を表現することができる．$STT_{sur}$ の単一使用制約は，$STT_{sur}$ が 単項二階述語論理による木変換器 (MSOTT) と同等の表現力となるために付けられている制約である．この単一使用制約がないストリーム木変換器の表現力についてはこれまで議論されておらず，その等価性判定が決定可能か否かは未解決問題である．本論文では $STT_{sur}$ の単一使用制約をなくし，入力をランク付きの木を表現するネスト文字列，出力を文字列に制限した決定性ストリーム変換器 (SRTST) を考える．また，SRTST とは別に木を入力とし文字列を出力する変換を扱う計算モデルとして正規先読み付き木から文字列への決定性下降型変換器 ($yDT^R$) がある．$yDT^R$ は累積引数を伴わない構造的な再帰関数を扱うことができる．$yDT^R$ の等価性判定は決定可能であることが証明されている．本論文では，SRTST から 等価な $yDT^R$ ，$yDT^R$ から等価な SRTST を構成する方法を示すことにより，SRTST と $yDT^R$ が同等の表現力を持つことを示す．この証明での SRTST から $yDT^R$ への変換は構成的であるため，$yDT^R$ の等価性判定を利用することにより，SRTST の等価性判定問題も決定可能となる．

平成 30 年度修士論文

# ランク付き木から文字列への<br>決定性ストリーム変換器の表現力

電気通信大学
大学院情報理工学研究科
情報・ネットワーク工学専攻
コンピュータサイエンスプログラム

学籍番号　：1731095
氏名　：高橋 祐多
主任指導教員　：岩崎 英哉 教授
指導教員　：村尾 裕一 准教授
提出日　：2019 年 3 月 14 日

## 要旨

構造化されたデータに対する変換は，関数プログラミングや文書処理において重要な役割を果しており，そのような変換のふるまいのよいクラスを明らかにするため，様々な計算モデルが提案されている．そのひとつとして，Alur と D'Antoni らによって提案された単一使用制約付き決定性ストリーム木変換器 ($\mathrm{STT}_{\mathrm{sur}}$) がある．$\mathrm{STT}_{\mathrm{sur}}$ の単一使用制約は，$\mathrm{STT}_{\mathrm{sur}}$ が 単項二階述語論理による木変換器 (MSOTT) と同等の表現力となるために付けられている制約である．この単一使用制約がないストリーム木変換器の表現力についてはこれまで議論されておらず，その等価性判定が決定可能か否かは未解決問題である．本論文では $\mathrm{STT}_{\mathrm{sur}}$ の単一使用制約をなくし，入力をランク付きの木を表現するネスト文字列，出力を文字列に制限した決定性ストリーム変換器 (SRTST) を考える．また，SRTST とは別に木を入力とし文字列を出力する変換を扱う計算モデルとして正規先読み付き木から文字列への決定性下降型変換器 ($\mathrm{yDT}^R$) がある．$\mathrm{yDT}^R$ は累積引数を伴わない構造的な再帰関数を扱うことができる．$\mathrm{yDT}^R$ の等価性判定は決定可能であることが証明されている．本論文では SRTST と $\mathrm{yDT}^R$ が同等の表現力を持つことを示す．この証明での SRTST から $\mathrm{yDT}^R$ への変換は構成的であるため，$\mathrm{yDT}^R$ の等価性判定を利用することにより，SRTST の等価性判定問題も決定可能となる．

# 目次

# 1　はじめに

## 1.1　背景

構造化されたデータに対する変換は，関数プログラミングや文書処理において重要な役割を果している．そのような変換に対して，ふるまいのよいクラスを明らかにするためさまざまな計算モデルが提案されている (ふるまいのよさとは，等価性判定や型検査が決定可能かどうかやクラスが合成で閉じているかなどである) [24, 16, 15, 13, 10, 20]．そのひとつとして，Alur と D'Antoni らによって提案された**単一使用制約付きストリーム木変換器** (*streaming tree transducer with single-use-restriction*，$\mathrm{STT}_{\mathrm{sur}}$) がある [2] [*1]．$\mathrm{STT}_{\mathrm{sur}}$ は**可視プッシュダウンオートマトン** (*visibly pushdown automaton*) と**ストリーム文字列変換器** (*streaming string transducer*) の特徴を合わせたようなモデルであり，ネスト文字列上の変換を定義することができる [3, 14, 2]．ここで**ネスト文字列**とは，**呼び出し/戻り** (*call/return*) でタグ付けされたアルファベット上の文字列であり，木や森などの階層構造を表現することができる．$\mathrm{STT}_{\mathrm{sur}}$ は入力されたネスト文字列を左から読み取り，入力記号，現在の状態，スタックのトップに積まれた値に従って有限個の変数 (レジスタ) の更新とスタックに関する操作を行い，出力するネスト文字列を構成する．変数の更新では，文字列の連結や木の挿入といった操作を行うことができる．入力を左から右に一度だけ読み取るため，入力を一旦すべて読み込まずとも計算を進めることができる．また，変換はスタックと変数に対する操作により定義されることから，同等の表現力を持つ他の変換器に比べて実装に適したモデルとなっている．$\mathrm{STT}_{\mathrm{sur}}$ は型検査や関数の等価性判定が決定可能である．

$\mathrm{STT}_{\mathrm{sur}}$ は **単一使用制約** (*single-use-restriction*) と呼ばれる制約が変数の更新にかけられているが，これは出力の計算において，各変数の値が本質的に高々一回のみ最終出力に寄与するということを保証する制約である．これにより，$\mathrm{STT}_{\mathrm{sur}}$ は**単項二階述語論理による木変換器** (*monadic second order definable tree transducer*，MSOTT) と同等の表現力を持つ [2]．この表現力の対応は，二つの変換が関数として等しいかを判定する**等価性判定問題**の解決にも寄与している．つまり，MSOTT の等価性判定が決定可能である [11] ことから，$\mathrm{STT}_{\mathrm{sur}}$ の等価性判定も決定可能となる．しかし，単一使用制約がない場合の表現力についてはこれまで議論されておらず，その等価性判定が決定可能か否かは未解決問題であった [2]．

本論文では，単一使用制約のない決定性ストリーム変換器のうち，入力をランク付き木を表現するネスト文字列，出力を文字列に制限したものを考える．そのような変換器を**ランク**

[*1] [2] では，単一使用制約が付いた変換器を**ストリーム木変換器**と呼んでいる．しかし，本論文では単一使用制約を伴わない場合を扱うため，明示的に区別して**単一使用制約付きストリーム木変換器**と呼ぶ．


図 1.1　構造化されたデータから JSON への変換

```
⟨books
  ⟨book
    ⟨The Art of Computer Programming⟩
    ⟨D.E.Knuth⟩
  book⟩
  ⟨empty⟩
books⟩
```

図 1.2　ネスト文字列による構造化されたデータの表現

**付き木から文字列への決定性ストリーム変換器** (*deterministic streaming ranked-tree-to-string transducer*，SRTST) と呼ぶこととする．SRTST はネスト文字列 (木) を入力として文字列を出力する変換を形式的に扱うモデルのひとつであり，$\mathrm{STT}_{\mathrm{sur}}$ と同様に入力されたネスト文字列を左から右に一回だけ走査し，出力文字列を構成する．SRTST は，図 1.1 に示すような構造化されたデータから文字列への変換を扱うことができるが，入力は木が直接与えられるのではなく，図 1.2 のような木を表現するネスト文字列として与えられる．

SRTST とは別に，木を入力として文字列を出力する変換器として**木から文字列への決定性下降型変換器** (*deterministic top-down tree-to-string transducer*，yDT) [*2] がある．yDT は累積引数を伴わない構造的な再帰関数を表現することができる．例として，図 1.1 の関数 *toJSON*

[*2] yDT などの文献では，ランク付き木を木と呼び，ランク無し木を森と呼んでいるが，$\mathrm{STT}_{\mathrm{sur}}$ などの文献では，ランク付き木をランク付き木と呼び，ランク無し木を木と呼んでいる．本論文ではそれぞれの命名法に合わせてあるが，本論文で扱う木はすべてランク付き木である．

を yDT の形式で書くと以下のようになる．

$$
\begin{aligned}
&\mathit{toJSON}(t) && = \texttt{[}\mathit{books}(t)\texttt{]}\\
&\mathit{books}(\texttt{books}(x_1, x_2)) && = \{\mathit{info}(x_1)\}\mathit{rest}(x_2)\\
&\mathit{books}(\texttt{empty}) && = \varepsilon\\
&\mathit{rest}(\texttt{books}(x_1, x_2)) && = \texttt{,}\{\mathit{info}(x_1)\}\mathit{rest}(x_2)\\
&\mathit{rest}(\texttt{empty}) && = \varepsilon\\
&\mathit{info}(\texttt{book}(x_1, x_2)) && = \texttt{title:"}\mathit{id}(x_1)\texttt{",author:"}\mathit{id}(x_2)\texttt{"}
\end{aligned}
$$

yDT に対して**正規先読み** (*regular look-ahead*) と呼ばれる入力された木の形に応じて行う変換を決定する拡張を加えたものを**正規先読み付き木から文字列への決定性下降型変換器** (*deterministic top-down tree-to-string transducer with regular look-ahead*, $\mathrm{yDT}^R$) [9] という．例えば，図 1.1 の関数 *toJSON* は $\mathrm{yDT}^R$ を用いると次の関数 *toJSON*′ として定義することができる．

$$
\begin{aligned}
&\mathit{toJSON}'(t) && = \texttt{[]} && \text{if } t = \texttt{empty}\\
&\mathit{toJSON}'(t) && = \texttt{[}\mathit{books}'(t)\texttt{]}\\
&\mathit{books}'(\texttt{books}(x_1, x_2)) && = \{\mathit{info}(x_1)\} && \text{if } x_2 = \texttt{empty}\\
&\mathit{books}'(\texttt{books}(x_1, x_2)) && = \{\mathit{info}(x_1)\}\texttt{,}\mathit{books}'(x_2)
\end{aligned}
$$

この例では，$t = \texttt{empty}$ や $x_2 = \texttt{empty}$ が正規先読みにあたる．ここで示した関数 *toJSON*′ と等価な関数 (関数 *toJSON*) を yDT により定義することができるが，一般に $\mathrm{yDT}^R$ は yDT より真に表現力が強い．また，$\mathrm{yDT}^R$ の等価性判定は決定可能である [22] ことが知られている．つまり，任意の二つの $\mathrm{yDT}^R$ が与えられたときそれらが関数として等しい変換であるか決定することができる．しかし，$\mathrm{yDT}^R$ は入力が与えられるとき入力すべてを読む必要があるのに加え，先読みを実現するため部分木に対して何度も走査する必要がある．一方，先に触れたように SRTST は入力木 (を表すネスト文字列) を一度走査するだけで出力することができる．

## 1.2 目的と方針

本論文では，SRTST と $\mathrm{yDT}^R$ が同等の表現力を持つこと示す．方針は，任意の SRTST から等価な $\mathrm{yDT}^R$ ，及び，任意の $\mathrm{yDT}^R$ から等価な SRTST を構成するそれぞれの方法を示すことにより，正当性を主張するものである．その結果，$\mathrm{yDT}^R$ の等価性判定が決定可能であることから SRTST の等価性判定も決定可能であることが導かれる．

## 1.3 本論文の構成

本論文の構成は次のとおりである．2 章では，文字列や木，ネスト文字列に関する基本的な定義を述べた後，非決定性上昇型木オートマトンを導入する．3 章では，正規先読み付き木から文

字列への決定性下降型変換器の定義を示す．4 章では，木から文字列への決定性ストリーム変換器の定義を示す．5 章では，はじめに任意の $\mathrm{yDT}^R$ から等価な変換を行う SRTST を構築する方法とその正当性を示した後，示した構成法により SRTST を構成する例を見る．その後，任意の SRTST から等価な変換を行う $\mathrm{yDT}^R$ を構築する方法とその正当性を示した後，示した構成法により $\mathrm{yDT}^R$ を構成する例を見る．最後に，構成法からわかることについて述べる．6 章 で関連研究について述べ，SRTST や $\mathrm{yDT}^R$ との関係について示す．7 章 で本論文のまとめと今後の研究課題について述べる．

# 2 準備

本章では，文字列，木，ネスト文字列について基本的な用語と概念の導入を行った後，**非決定性上昇型木オートマトン** (BTA) の定義を示す．

空集合を表す記号として $\emptyset$ を用いる．0 を含む自然数の集合を $\mathbb{N}$ で表す．自然数の集合 $\{1,\ldots,n\}$ を $[n]$ と表す．特に $n=0$ のとき $[n]=\emptyset$ である．集合 $S$ について $S$ の要素の数を $|S|$ で表す．集合 $S$，$S'$ について $S$ から $S'$ を引いた差集合を $S \setminus S'$ で表す．部分集合 $S \subseteq S'$ に対し，$S'$ が文脈上明らかなとき $S' \setminus S$ を単に $S^{\complement}$ で表す．集合 $\Delta$ の $k$ 個の要素を並べた $k$-タプルすべてからなる集合を $\Delta^k$ と表す．特に $k=0$ のとき $\Delta^k$ は空のタプル $()$ のみを含む集合となる．$k$-タプル $\boldsymbol{t}$ の $i$ 番目の要素を $\boldsymbol{t}_i$ で表す．$k$-タプル $\boldsymbol{t}$ について $|\boldsymbol{t}|$ で要素数 $k$ を表す．$k$-タプル $\boldsymbol{t} \in \Delta^k$ と $d \in \Delta$ が与えられたとき，$k$-タプルの最後の要素に $d$ を加えた $k+1$-タプル $(\boldsymbol{t}_1,\ldots,\boldsymbol{t}_k,d)$ を $\boldsymbol{t}\,||\,d$ と表す．$\Delta$ の $k$ 個以下の要素を持つタプルすべてからなる集合を $\Delta^{\leq k} = \bigcup_{i=0}^{k} \Delta^i$ とする．空でない有限集合を**アルファベット**と呼ぶ．アルファベットの要素を**シンボル**または**記号**と呼ぶ．$A$ から $B$ への部分関数 $f$ を $f : A \rightharpoonup B$ と表し，$f$ の定義域を $\mathrm{Dom}(f)$ で表す．

## 2.1 文字列，木，ネスト文字列

$\Delta$ をアルファベットとするとき，有限個の $\Delta$ の要素 $w_1,\ldots,w_n$ の列 $w_1 \cdots w_n$ を $\Delta$ **上の文字列**と呼ぶ．特に長さ 0 の文字列を $\varepsilon$ で表し，**空文字列**と呼ぶ．文字列 $u = u_1 \cdots u_n$，$v = v_1 \cdots v_m$ について，その連接 '$\cdot$' を $u \cdot v = u_1 \cdots u_n v_1 \cdots v_m$ と定義する．また '$\cdot$' を省略し，単に $uv$ とも書く．$\Delta$ 上の文字列すべてからなる集合を $\Delta^*$ と表す．$\Delta^*$ の部分集合 $L$ を**言語**という．

**ランク付きアルファベット**とは，アルファベット $\Sigma$ と，$\Sigma$ の各シンボルに**ランク**と呼ばれる自然数を対応付ける関数 $\mathrm{rank}_\Sigma : \Sigma \to \mathbb{N}$ からなる組 $(\Sigma, \mathrm{rank}_\Sigma)$ である．ランクが $k$ のランク付きシンボル $\sigma$ を $\sigma^{(k)}$ と表し，ランク付きアルファベット $\Sigma$ に含まれるランクが $k$ のシンボルからなる集合を $\Sigma^{(k)}$ と表す．また，$\Sigma$ に含まれるランク付きシンボルのうち，ランクが $k$ より大きいシンボルからなる集合を $\Sigma^{(>k)}$ と表す．可算無限集合 $X = \{x_1, x_2, \ldots\}$ を任意に固定し，その元を**入力変数**と呼ぶ．ある自然数 $k \in \mathbb{N}$ について $X_k = \{x_1,\ldots,x_k\}$ とする．特に $k=0$ のとき $X_k = \emptyset$ とする．

**定義 1.** **ランク付きアルファベット $\Sigma$ 上の木**の集合 $\mathcal{T}_\Sigma$ は，すべての $\sigma^{(n)} \in \Sigma$ について $t_1,\ldots,t_n \in \mathcal{T}_\Sigma$ ならば $\sigma(t_1,\ldots,t_n) \in \mathcal{T}_\Sigma$ を満たす最小の集合である． ■

$e \in \Sigma^{(0)}$ について $e() \in \mathcal{T}_\Sigma$ を単に $e$ と書く．木の定義からわかるように，ランク付きアル

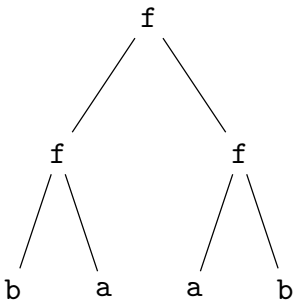


図 2.1　木 f(f(b,a),f(a,b))

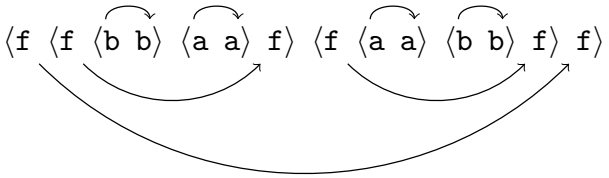


図 2.2　ネスト文字列 ⌊f(b,a),f(a,b)⌋

ファベットのランクは各ノードのシンボルが持つことのできる子供の数を表している．

$\Sigma$ をアルファベットとするとき，$\hat{\Sigma}$ を**タグ付きアルファベット**と呼び，すべての $\sigma \in \Sigma$ について「$\langle\sigma$」と「$\sigma\rangle$」の二つのシンボルを含む集合とする (すなわち $\hat{\Sigma} = \bigcup_{\sigma\in\Sigma} \{\langle\sigma\} \cup \{\sigma\rangle\}$) [*1]．$\hat{\Sigma}$ 上の文字列を**ネスト文字列**と呼ぶ．$\sigma \in \Sigma$ について $\langle\sigma$ を**呼出し記号**，$\sigma\rangle$ を**戻り記号**と呼ぶ．$\sigma$ の違いを無視して呼び出しと戻りの対応が取れたネスト文字列を**整合的** (*well-matched*) という．さらに各 $\sigma \in \Sigma$ について，「$\langle\sigma$」と「$\sigma\rangle$」の対応が取れたネスト文字列を**整形式** (*well-formed*) という．整形式ならば常に整合的である．

**例 1.** ランク付きアルファベットを $\Sigma = \{\mathtt{f}^{(2)}, \mathtt{a}^{(0)}, \mathtt{b}^{(0)}\}$ とする．

$$t = \mathtt{f(f(b,a),f(a,b))} \in \mathcal{T}_\Sigma$$

は，$\mathcal{T}_\Sigma$ の要素であり，図 2.1 のように図示される．しかし，次のような $t'$ はランク 0 のシンボル b が子供を持っているため $\mathcal{T}_\Sigma$ の要素でない．

$$t' = \mathtt{f(b(a),f(b,a))} \notin \mathcal{T}_\Sigma$$

次のようなネスト文字列を考える．

$$\langle \mathtt{f}\ \langle \mathtt{b}\ \mathtt{b}\rangle\ \langle \mathtt{a}\ \mathtt{a}\rangle\ \mathtt{f}\rangle\ \langle \mathtt{f}\ \langle \mathtt{a}\ \mathtt{a}\rangle\ \langle \mathtt{b}\ \mathtt{b}\rangle\ \mathtt{f}\rangle\ \mathtt{f}\rangle \tag{2.1}$$

$$\langle \mathtt{f}\ \langle \mathtt{f}\ \langle \mathtt{b}\ \mathtt{b}\rangle\ \langle \mathtt{a}\ \mathtt{b}\rangle\ \mathtt{f}\rangle\ \langle \mathtt{f}\ \langle \mathtt{a}\ \mathtt{a}\rangle\ \langle \mathtt{b}\ \mathtt{b}\rangle\ \mathtt{f}\rangle \tag{2.2}$$

$$\langle \mathtt{f}\ \langle \mathtt{a}\ \mathtt{a}\rangle\ \langle \mathtt{b}\ \mathtt{b}\rangle\ \langle \mathtt{a}\ \mathtt{a}\rangle\ \mathtt{f}\rangle \tag{2.3}$$

式 (2.1) は，戻り記号 f⟩ に対応する呼び出し記号が存在しないため，整合的でないネスト文字列である．式 (2.2) は整合的であるが，呼び出し記号 ⟨a と戻り記号 b⟩ が正しく対応しておらず整形式でない．式 (2.3) は，整形式であるがランク 2 のシンボル f が三つ子供を持つため $\Sigma$ 上のランク付き木を表現していない． ■

[*1] 一般的にタグ付きアルファベットは call/internal/return の 3 つの区別可能なタグ付きシンボルからなる集合として定義されるが，本論文ではランク付き木を表現するネスト文字列のみを扱うため call/return だけからなる集合として定義している．

以下，$\Sigma$ をランク付きアルファベットとする．

**定義 2.** 木からネスト文字列への変換を行う関数 $\lfloor - \rfloor : \mathcal{T}_\Sigma \to \hat{\Sigma}^*$ を次のように定義する．すべての $t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$ について，

$$\lfloor t \rfloor = \langle \sigma \ \lfloor t_1 \rfloor \cdots \lfloor t_n \rfloor \ \sigma \rangle$$

とする． ■

木 $\mathcal{T}_\Sigma$ から生成されるネスト文字列の集合を $\lfloor \mathcal{T}_\Sigma \rfloor$ とし，$\lfloor \mathcal{T}_\Sigma \rfloor$ の元を**ランク付きネスト文字列**という．ランク付きネスト文字列は整形式であるが，逆は必ずしも成り立たない．あとで見るように，本論文ではネスト文字列の変換器の入力としては，ランク付きネスト文字列のみを対象とする．$e \in \Sigma^{(0)}$ について，ネスト文字列 $\langle e \ e \rangle$ $(= \lfloor e \rfloor)$ を単に $\langle e \rangle$ と表す．

**定義 3.** ランク付きネスト文字列からランク付き木への変換を行う関数 $\lceil - \rceil : \lfloor \mathcal{T}_\Sigma \rfloor \to \mathcal{T}_\Sigma$ を次のように定義する．すべてのネスト文字列 $w = \langle \sigma^{(n)} \ w_1 \cdots w_n \ \sigma^{(n)} \rangle \in \lfloor \mathcal{T}_\Sigma \rfloor$ について，

$$\lceil w \rceil = \sigma(\lceil w_1 \rceil, \ldots, \lceil w_n \rceil)$$

とする．ただし，$w_1, \ldots, w_n \in \lfloor \mathcal{T}_\Sigma \rfloor$ とする． ■

$\lfloor - \rfloor$ と $\lceil - \rceil$ について，明らかにすべての $t \in \mathcal{T}_\Sigma$ で $t = \lceil \lfloor t \rfloor \rceil$ が成り立つ．また，すべての $w \in \lfloor T_\Sigma \rfloor$ について $w = \lfloor \lceil w \rceil \rfloor$ が成り立つ．

**例 2.** 例 1 に続き，$\Sigma = \{\mathtt{f}^{(2)}, \mathtt{a}^{(0)}, \mathtt{b}^{(0)}\}$ とする．木 `f(f(b,a),f(a,b))` をネスト文字列で表すと $w$ のようになる．

$$\begin{aligned} w &= \lfloor \mathtt{f(f(b,a),f(a,b))} \rfloor \\ &= \langle \mathtt{f} \ \langle \mathtt{f} \ \langle \mathtt{b} \ \mathtt{b} \rangle \ \langle \mathtt{a} \ \mathtt{a} \rangle \ \mathtt{f} \rangle \ \langle \mathtt{f} \ \langle \mathtt{a} \ \mathtt{a} \rangle \ \langle \mathtt{b} \ \mathtt{b} \rangle \ \mathtt{f} \rangle \ \mathtt{f} \rangle \\ &= \langle \mathtt{f} \ \langle \mathtt{f} \ \langle \mathtt{b} \rangle \ \langle \mathtt{a} \rangle \ \mathtt{f} \rangle \ \langle \mathtt{f} \ \langle \mathtt{a} \rangle \ \langle \mathtt{b} \rangle \ \mathtt{f} \rangle \ \mathtt{f} \rangle \in \lfloor \mathcal{T}_\Sigma \rfloor \end{aligned}$$

$w$ の呼び出し記号と戻り記号の対応は図 2.2 のように図示される．また，以下のネスト文字列 $w'$ は整形式であるが $\lfloor \mathcal{T}_\Sigma \rfloor$ には含まれない．

$$w' = \langle \mathtt{f} \ \langle \mathtt{b} \ \mathtt{b} \rangle \ \langle \mathtt{a} \ \mathtt{a} \rangle \ \mathtt{f} \rangle \ \langle \mathtt{f} \ \langle \mathtt{a} \ \mathtt{a} \rangle \ \langle \mathtt{b} \ \mathtt{b} \rangle \ \mathtt{f} \rangle \notin \lfloor \mathcal{T}_\Sigma \rfloor$$

$w'$ は二つの木 `f(b,a)`，`f(a,b)` の列を表している． ■

## 2.2 非決定性上昇型木オートマトン

本節では，**非決定性上昇型木オートマトン** (BTA) の定義と BTA が受理する言語について述べる．BTA は木を受理するオートマトンであり，BTA の受理する言語のクラスは**正規木言語** (*regular tree language*) のクラスと一致する．

**定義 4. 非決定性上昇型木オートマトン** (*non-deterministic bottom-up tree automaton*, BTA) は，次のように定義される組 $(\Pi, \Sigma, \theta)$ である．

- $\Pi$ は**状態**の有限集合，
- $\Sigma$ はランク付きアルファベット，
- $\theta : \Sigma^{(n)} \times \Pi^n \to 2^{\Pi}$ は**遷移関数**. ■

BTA $A = (\Pi, \Sigma, \theta)$ とする．遷移関数 $\theta$ の自然な拡張 $\hat{\theta} : \mathcal{T}_\Sigma \to 2^{\Pi}$ を次のように定義する．木 $\sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$ について，

$$\hat{\theta}(\sigma(t_1, \ldots, t_n)) = \bigcup_{\pi_1 \in \hat{\theta}(t_1), \ldots, \pi_n \in \hat{\theta}(t_n)} \theta(\sigma, \pi_1, \ldots, \pi_n)$$

とする．$\pi \in \Pi$, $t \in \mathcal{T}_\Sigma$ について $\pi \in \hat{\theta}(t)$ のとき，$\pi$ が $t$ を**受理する**という．$\pi$ が受理する木の集合を $\mathcal{L}_A(\pi) = \{t \in \mathcal{T}_\Sigma \mid \pi \in \hat{\theta}(t)\}$ と定義する．$A$ が明らかなとき単に $\mathcal{L}(\pi)$ と書く．$\pi \in \Pi$ がある木を受理するとき $\pi$ が**到達可能**であるという．すべての状態について到達可能なとき，$A$ は**到達可能条件**を満たすといい，本論文では到達可能条件を満たす BTA だけを考える．このとき，すべての $\pi \in \Pi$ について $\mathcal{L}(\pi) \neq \emptyset$ が成り立つ．

すべての $\sigma^{(n)} \in \Sigma$，$\pi_1, \ldots, \pi_n \in \Pi$ について $\theta(\sigma, \pi_1, \ldots, \pi_n)$ の要素数が高々 1 のとき，**決定性上昇型木オートマトン** (*determinisitc bottom-up tree autoamton*, DBTA) という．DBTA は，すべての $t \in \mathcal{T}_\Sigma$ について $|\hat{\theta}(t)| \leq 1$ であり，$\hat{\theta}(t) = \{\pi\}$ なる $\pi$ が存在するか，$\hat{\theta}(t) = \emptyset$ である．本論文では，DBTA について遷移関数 $\theta$ を $\Sigma^{(n)} \times \Pi^n$ から $\Pi$ への写像として扱う．ただし，$\hat{\theta}(t) = \emptyset$ のとき $\perp \notin \Pi$ なる状態に遷移するものとする．このとき，すべての $t \in \mathcal{T}_\Sigma$ について，少なくとも一つは $t$ を受理する状態が存在する．DBTA が受理する言語のクラス $\{\mathcal{L}_A(\pi) \mid A = (\Pi, \Sigma, \theta) : \text{DBTA}, \pi \in \Pi\}$ と BTA が受理する言語のクラス $\{\mathcal{L}_A(\pi) \mid A = (\Pi, \Sigma, \theta) : \text{BTA}, \pi \in \Pi\}$ は，どちらも正規木言語と同じクラスであり，和集合・積集合・補集合について閉じており，それらの言語を受理する BTA を構築することができる [7]．ここでは，積集合について簡単に述べる．$A = (\Sigma, \Pi, \theta)$，$A' = (\Sigma, \Pi', \theta')$ を BTA とするとき，$A$ と $A'$ の積オートマトンを $A \otimes A'$ で表し，$A \otimes A' = (\Sigma, \Pi \times \Pi', \theta_{A \otimes A'})$ と定義する．ここで，すべての $\sigma^{(n)} \in \Sigma$，$(\pi_1, \pi'_1), \ldots, (\pi_n, \pi'_n) \in \Pi \times \Pi'$ について，

$$\begin{aligned}&\theta_{A \otimes A'}(\sigma, (\pi_1, \pi'_1), \ldots, (\pi_n, \pi'_n))\\ &= \{(\pi, \pi') \in \Pi \times \Pi' \mid \pi \in \theta(\sigma, \pi_1, \ldots, \pi_n) \wedge \pi' \in \theta(\sigma, \pi'_1, \ldots, \pi'_n)\}\end{aligned}$$

とする．この定義により，すべての $(\pi, \pi') \in \Pi \times \Pi'$ について $\mathcal{L}_{A \otimes A'}((\pi, \pi')) = \mathcal{L}_A(\pi) \cap \mathcal{L}_{A'}(\pi')$ が成り立つ．

**例 3.** BTA $A = (\Pi, \Sigma, \theta)$ を次のように定義する．$\Pi = \{\pi_{\mathtt{a}}, \pi_{\mathtt{b}}\}$，$\Sigma = \{\mathtt{f}^{(2)}, \mathtt{a}^{(0)}, \mathtt{b}^{(0)}\}$ とし，遷移関数 $\theta$ を次のように与える．

$$\theta(\mathtt{f}, \pi_1, \pi_2) = \{\pi_1\}, \qquad \theta(\mathtt{a}, ()) = \{\pi_{\mathtt{a}}\}, \qquad \theta(\mathtt{b}, ()) = \{\pi_{\mathtt{b}}\}.$$

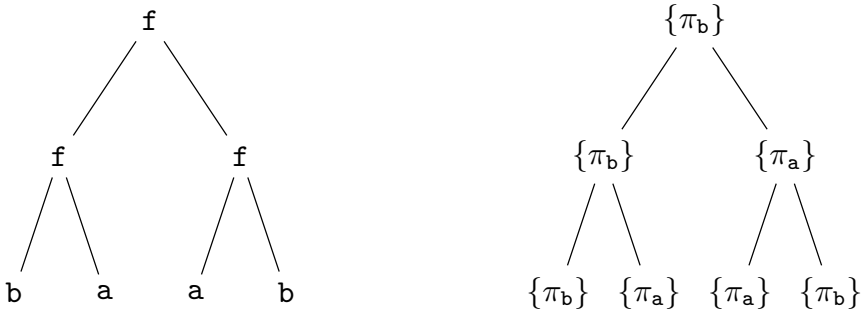


図 2.3　BTA の遷移関数による状態の割り当て

すべての $\sigma^{(n)} \in \Sigma$，$\pi_1, \ldots, \pi_n \in \Pi$ について $|\theta(\sigma, \pi_1, \ldots, \pi_n)| = 1$ であることから $A$ は DBTA である．$\mathcal{L}(\pi_{\mathtt{a}})$ は最も左側の葉が a となる木すべてを含む集合であり，$\mathcal{L}(\pi_{\mathtt{b}})$ は最も左側の葉が b となる木すべてを含む集合である．例えば $\hat{\theta}(\mathtt{a}) = \{\pi_{\mathtt{a}}\}$，$\hat{\theta}(\mathtt{b}) = \{\pi_{\mathtt{b}}\}$ から $\mathtt{a} \in \mathcal{L}(\pi_{\mathtt{a}})$，$\mathtt{b} \in \mathcal{L}(\pi_{\mathtt{b}})$ である．また，木 $\mathtt{f(b,a)}, \mathtt{f(a,b)} \in \mathcal{T}_\Sigma$ について，

$$\hat{\theta}(\mathtt{f(b,a)}) = \bigcup_{\pi_1 \in \hat{\theta}(\mathtt{b}), \pi_2 \in \hat{\theta}(\mathtt{a})} \theta(\mathtt{f}, \pi_1, \pi_2) = \{\pi_{\mathtt{b}}\}, \qquad \hat{\theta}(\mathtt{f(a,b)}) = \bigcup_{\pi_1 \in \hat{\theta}(\mathtt{a}), \pi_2 \in \hat{\theta}(\mathtt{b})} \theta(\mathtt{f}, \pi_1, \pi_2) = \{\pi_{\mathtt{a}}\}.$$

であるため，$\mathtt{f(b,a)} \in \mathcal{L}(\pi_{\mathtt{b}})$，$\mathtt{f(a,b)} \in \mathcal{L}(\pi_{\mathtt{a}})$ となる．同様にして $\mathtt{f(f(b,a),f(a,b))} \in \mathcal{L}(\pi_{\mathtt{b}})$ とわかる．

$$\hat{\theta}(\mathtt{f(f(b,a),f(a,b))}) = \bigcup_{\pi_1 \in \hat{\theta}(\mathtt{f(b,a)}), \pi_2 \in \hat{\theta}(\mathtt{f(a,b)})} \theta(\mathtt{f}, \pi_1, \pi_2) = \{\pi_{\mathtt{b}}\}.$$

$\hat{\theta}(\mathtt{f(f(b,a),f(a,b))})$ の計算において，各ノードに割り当てられる状態は図 2.3 のようになる．

■

# 3　先読み付き木から文字列への決定性下降型変換器

本章では，**正規先読み付き木から文字列への決定性下降型変換器** ($\mathrm{yDT}^R$) の定義とその具体例を示す．$\mathrm{yDT}^R$ は**木から文字列への決定性下降型変換器** (yDT) に対して**正規先読み** (*regular look-ahead*) と呼ばれる拡張をしたものである．yDT は入力として木を受け取り，文字列を出力する関数を扱う計算モデルであり，累積引数を伴わない構造的な再帰関数を定義することができる．

はじめに $\mathrm{yDT}^R$ の規則の右辺で用いる表現の集合を定義する．

**定義 5.** (状態の) 有限集合 $Q$，アルファベット $\Delta$，変数集合 $X_n$ $(n \in \mathbb{N})$ について，次の文法から生成される列すべてを含む集合を $\mathrm{Rhs}_{Q,\Delta}(X_n)$ と表す．

$$\tau ::= \varepsilon \mid a\tau \mid q(x_i)\tau$$

ここで $a \in \Delta$，$q \in Q$，$i \in [n]$ とする．また，$q(x_i)$ はタプル $(q, x_i)$ を表す． ■

集合 $\mathrm{Rhs}_{Q,\Delta}(X_n)$ は $\mathrm{yDT}^R$ の初期列と変換規則の定義に用いる．$\tau \in \mathrm{Rhs}_{Q,\Delta}(X_n)$ について $\tau$ に $q(x_i)$ が現れるとき $q(x_i) \in \tau$ と書く．

$\mathrm{yDT}^R$ の形式的な定義を以下に示す．比較的新しい文献 [12, 18] では，正規先読みには DBTA が用いられるが，SRTST から $\mathrm{yDT}^R$ を構築するとき，先読みオートマトンが BTA である方が都合が良いため，ここでは BTA を正規先読みとして用いるが，どちらも表現力は等しい．BTA の遷移は非決定的であるが，各変換規則は決定的である点に注意する必要がある．

**定義 6. 正規先読み木から文字列への決定性下降型変換器** (*deterministic top-down tree-to-string transducer with regular look-ahead*, $\mathrm{yDT}^R$) は，次のように定義される組 $(Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ である．

- $Q$ は**状態**の有限集合，
- $\Sigma$ はランク付き**入力**アルファベット，
- $\Delta$ は**出力**アルファベット，
- $(\Pi, \Sigma, \theta)$ は BTA であり，**先読みオートマトン**と呼ぶ．
- $\mathit{Init} \subseteq \mathrm{Rhs}_{Q,\Delta}(X_1) \times \Pi$ は**初期列**と先読みオートマトンの状態の組の集合であり，すべての $(\tau, \pi) \neq (\tau', \pi') \in \mathit{Init}$ について $\mathcal{L}(\pi) \cap \mathcal{L}(\pi') = \emptyset$ とする．$(\Pi, \Sigma, \theta)$ は到達可能条件を満たすため $(\tau, \pi) \in \mathit{Init}$ となる $\pi$ は高々一つであり，存在するとき $\tau$ を $\mathit{Init}(\pi)$ と表す．
- $R \subseteq \bigcup_{n \in \mathbb{N}} (Q \times \Sigma^{(n)} \times \mathrm{Rhs}_{Q,\Delta}(X_n) \times \Pi^n)$ は**正規先読み付き変換規則**の有限集合であ

り，各変換規則は次のような形で表す．

$$q(\sigma(x_1,\ldots,x_n)) \to \tau \qquad \langle \pi_1,\ldots,\pi_n \rangle$$

ここで $q \in Q$，$\sigma^{(n)} \in \Sigma$，$\tau \in \mathrm{Rhs}_{Q,\Delta}(X_n)$，$\pi_1,\ldots,\pi_n \in \Pi$ である．$\mathrm{rank}(\sigma) = 0$ のとき，$\sigma()$ を $\sigma$ と書き，$\langle\rangle$ を省略する．また，各変換規則は次のような意味で決定的であるとする．すべての $(q,\sigma,\tau,(\pi_1,\ldots,\pi_n)) \neq (q,\sigma,\tau',(\pi'_1,\ldots,\pi'_n)) \in R$ について，ある $i \in [n]$ が存在して $\pi_i \neq \pi'_i$ かつ $\mathcal{L}(\pi_i) \cap \mathcal{L}(\pi'_i) = \emptyset$ である．
$(\Pi,\Sigma,\theta)$ は到達可能条件を満たすため，$q \in Q$，$\sigma^{(n)} \in \Sigma$，$\pi_1,\ldots,\pi_n \in \Pi$ に対する変換規則は高々一つであり，それを $(q,\sigma,\pi_1,\ldots,\pi_n)$-規則と呼ぶ．$\boldsymbol{\pi} = (\pi_1,\ldots,\pi_n) \in \Pi^n$ とするとき $(q,\sigma,\pi_1,\ldots,\pi_n)$ の意味で $(q,\sigma,\boldsymbol{\pi})$ と表記する．$(q,\sigma,\boldsymbol{\pi})$-規則が存在するとき，$(q,\sigma,\boldsymbol{\pi}) \in R$ と表す．$(q,\sigma,\boldsymbol{\pi})$-規則の右辺 $\tau$ を $\mathrm{rhs}(q,\sigma,\boldsymbol{\pi})$ と表す． ■

$M = (Q,\Sigma,\Delta,\mathit{Init},R,\Pi,\theta)$ を $\mathrm{yDT}^R$ とする．相互再帰により，状態 $q \in Q$ に対し部分関数 $[\![q]\!]_M : \mathcal{T}_\Sigma \rightharpoonup \Delta^*$ を，そして $\tau \in \mathrm{Rhs}_{Q,\Delta}(X_n)$ に対し部分関数 $[\![\tau]\!]_M : \mathcal{T}_\Sigma^n \rightharpoonup \Delta^*$ を次のように定義する．木 $\sigma(t_1,\ldots,t_n) \in \mathcal{T}_\Sigma$ について，ある $\pi_1,\ldots,\pi_n \in \Pi$ が存在して $t_1 \in \mathcal{L}(\pi_1),\ldots,t_n \in \mathcal{L}(\pi_n)$ かつ $(q,\sigma,\pi_1,\ldots,\pi_n)$-規則が定義されているとき，

$$[\![q]\!]_M(\sigma(t_1,\ldots,t_n)) = [\![\mathrm{rhs}(q,\sigma,\pi_1,\ldots,\pi_n)]\!]_M(t_1,\ldots,t_n)$$

とする (上の定義はよく定義されている，すなわち $\pi'_1,\ldots,\pi'_n$ も同様の条件を満たすとき $R$ の決定性から $\mathrm{rhs}(q,\sigma,\pi_1,\ldots,\pi_n) = \mathrm{rhs}(q,\sigma,\pi'_1,\ldots,\pi'_n)$ が言える)．部分関数 $[\![\tau]\!]_M : \mathcal{T}_\Sigma^n \rightharpoonup \Delta^*$ は次のように定義する．

$$\begin{aligned}
[\![\varepsilon]\!]_M(t_1,\ldots,t_n) &= \varepsilon \\
[\![a\tau]\!]_M(t_1,\ldots,t_n) &= a\,[\![\tau]\!]_M(t_1,\ldots,t_n) \\
[\![q'(x_i)\tau]\!]_M(t_1,\ldots,t_n) &= [\![q']\!]_M(t_i) \cdot [\![\tau]\!]_M(t_1,\ldots,t_n)
\end{aligned}$$

ただし， $a \in \Delta$，$q' \in Q$，$x_i \in X_n$ である．

以上の定義から，$M$ に対して部分関数 $[\![M]\!] : \mathcal{T}_\Sigma \rightharpoonup \Delta^*$ を以下のように定義する．$t \in \mathcal{T}_\Sigma$ に対してある $\pi \in \Pi$ が存在して $t \in \mathcal{L}(\pi)$ のとき $[\![M]\!](t) = [\![\mathit{Init}(\pi)]\!]_M(t)$ と定義し，そのような $\pi$ が存在しないならば未定義とする．($\mathit{Init}$ に関する条件から $\pi$ は一意である．) $[\![M]\!]$ の定義域を $\mathrm{Dom}(M)$ で表し，$[\![q]\!]_M$ の定義域を $\mathrm{Dom}(q)$ と表す．

**例 4.** 例 3 で用いた $\Pi$，$\theta$ を使って $\mathrm{yDT}^R$ $M = (Q,\Sigma,\Delta,\mathit{Init},R,\Pi,\theta)$ を次のように定義する．$Q = \{q\}$，$\Sigma = \{\mathtt{f}^{(2)},\mathtt{a}^{(0)},\mathtt{b}^{(0)}\}$，$\Delta = \{\mathtt{a},\mathtt{b}\}$，$\mathit{Init} = \{(q(x_1),\pi_\mathtt{a}),(q(x_1),\pi_\mathtt{b})\}$ とし，$R$ に含まれる変換規則を次のようにする．

$$\begin{array}{llllllll}
q(\mathtt{f}(x_1,x_2)) \to q(x_1)q(x_2) & \langle \pi_\mathtt{a},\pi_\mathtt{a}\rangle, & q(\mathtt{f}(x_1,x_2)) \to q(x_2)q(x_1) & \langle \pi_\mathtt{b},\pi_\mathtt{a}\rangle, & q(\mathtt{a}) \to \mathtt{a}, \\
q(\mathtt{f}(x_1,x_2)) \to q(x_1)q(x_2) & \langle \pi_\mathtt{a},\pi_\mathtt{b}\rangle, & q(\mathtt{f}(x_1,x_2)) \to q(x_2)q(x_1) & \langle \pi_\mathtt{b},\pi_\mathtt{b}\rangle, & q(\mathtt{b}) \to \mathtt{b}.
\end{array}$$

このとき，$[\![M]\!]$ は与えられた木について，一番左下の子供が $\mathtt{b}$ のとき左の子供と右の子供を交換するといった操作を子供に対して繰り返し行うことで得られる木の葉を左から右に出力する変換である．入力として $\mathtt{f(f(b,a),f(a,b))}$ が与えられたとき，次のようにして計算し，最終的な出力として $[\![M]\!]\,(\mathtt{f(f(b,a),f(a,b))}) = \mathtt{abab}$ を得る．

$$\begin{aligned}
[\![M]\!]\,(\mathtt{f(f(b,a),f(a,b))}) &= [\![\mathit{Init}(\pi_\mathtt{b})]\!]_M\,(\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![q(x_1)]\!]_M\,(\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![q]\!]_M\,(\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![\mathrm{rhs}(q,\mathtt{f},\langle\pi_\mathtt{b},\pi_\mathtt{a}\rangle)]\!]_M\,(\mathtt{f(b,a)},\mathtt{f(a,b)}) \\
&= [\![q(x_2)q(x_1)]\!]_M\,(\mathtt{f(b,a)},\mathtt{f(a,b)}) \\
&= [\![q]\!]_M\,(\mathtt{f(a,b)})\,[\![q]\!]_M\,(\mathtt{f(b,a)}) \\
&= [\![\mathrm{rhs}(q,\mathtt{f},\langle\pi_\mathtt{a},\pi_\mathtt{b}\rangle)]\!]_M\,(\mathtt{f(a,b)})\,[\![\mathrm{rhs}(q,\mathtt{f},\langle\pi_\mathtt{b},\pi_\mathtt{a}\rangle)]\!]_M\,(\mathtt{f(b,a)}) \\
&= [\![q(x_1)q(x_2)]\!]_M\,(\mathtt{a},\mathtt{b})\,[\![q(x_2)q(x_1)]\!]_M\,(\mathtt{b},\mathtt{a}) \\
&= [\![q]\!]_M\,(\mathtt{a})\,[\![q]\!]_M\,(\mathtt{b})\,[\![q]\!]_M\,(\mathtt{a})\,[\![q]\!]_M\,(\mathtt{b}) \\
&= \mathtt{abab}
\end{aligned}$$ ■

$M = (Q,\Sigma,\Delta,\mathit{Init},R,\Pi,\theta)$ を $\mathrm{yDT}^R$ とし，$A = (\Sigma,\Pi,\theta)$ とする．$M$ の定義から $\mathrm{Dom}(M) \subseteq \bigcup_{(\tau,\pi)\in\mathit{Init}} \mathcal{L}_A(\pi)$ が言える．$\mathrm{yDT}^R$ の定義域 $\mathrm{Dom}(M)$ は BTA によって表すことができるので，$\mathrm{Dom}(M) = \mathcal{L}_{A_0}(\pi_0)$ とし，$A_0 = (\Sigma,\Pi_0,\theta_0)$ とする．また，BTA は共通部分について閉じているため，$\mathcal{L}_{A\otimes A_0}(\pi_0) = \mathcal{L}_A(\pi) \cap \mathrm{Dom}(M)$ であり，$A' = (\Sigma,\Pi\times\Pi_0,\theta') \stackrel{\mathrm{def}}{=} A\otimes A_0$ とする．新しく $\mathrm{yDT}^R$ $M' = (Q,\Sigma,\Delta,\mathit{Init}',R',\Pi\times\Pi_0,\theta')$ を定義する．$\mathit{Init}' = \{(\tau,(\pi,\pi_0)) \mid (\tau,\pi)\in\mathit{Init}\}$ とし，$R'$ を以下で定義する．

$$q(\sigma(x_1,\ldots,x_n)) \to \mathrm{rhs}(q,\sigma,\pi_1,\ldots,\pi_n) \qquad \langle(\pi_1,\pi_1'),\ldots,(\pi_n,\pi_n')\rangle$$

このとき，$[\![M]\!] = [\![M']\!]$ でありかつ，

$$\mathrm{Dom}(M') = \biguplus_{(\tau',\pi')\in\mathit{Init}'} \mathcal{L}_{A'}(\pi') \tag{3.1}$$

が成り立つ．これ以降考える $\mathrm{yDT}^R$ は式 (3.1) を満たすものとする．また，同様に $\mathrm{Dom}(q)$ についても

$$\mathrm{Dom}(q) = \biguplus_{(q,\sigma,\boldsymbol{\pi})\in R} \{\sigma(t_1,\ldots,t_n) \mid t_1\in\mathcal{L}(\boldsymbol{\pi}_1),\ldots,t_n\in\mathcal{L}(\boldsymbol{\pi}_n)\}$$

が成り立つものとする．

# 4　木から文字列への決定性ストリーム変換器

本章では，**ランク付き木から文字列への決定性ストリーム変換器** (SRTST) の定義と具体例を示す．SRTST は，Alur と D'Antoni らによって提案された単一使用制約付きストリーム木変換器 ($\mathrm{STT}_{\mathrm{sur}}$) から単一使用制約をなくし，入力をランク付きネスト文字列，出力を文字列に制限したストリーム木変換器である．SRTST は，有限個の変数とスタックを持つ計算モデルである．有限個の変数には文字列が割り当てられ，各変数に対して割当てられた文字列を返す関数をここでは**評価**と呼ぶ．スタックにはスタック記号と評価の組が積まれる．SRTST は入力されたネスト文字列を左から右に走査し，状態，スタック，有限個の変数を更新していくことで出力文字列を構成する．呼び出し記号を読んだときは，状態の遷移と評価の更新をし，スタックにスタック記号と更新関数で定まる割当てにより計算された評価を積む．このとき，現在の評価をすべての変数について空文字列を返す評価に更新される．戻り記号を読んだときは，状態の遷移とスタックに積まれたスタック記号に従い状態を遷移し，更新関数で定まる割当てをスタックに積まれた評価とその時点の評価から計算し，その結果得られた評価で現在の評価を更新する．最後に，入力を読み終えたときの状態と評価に従い文字列を出力する．

## 4.1　評価と割当て

次節での SRTST の定義のために，ここでは**評価**や評価を更新するための**表現**，評価を更新するための写像 (**割当て**) の導入と，それらに関するいくつかの表記法を定義する．

**変数**の有限集合 $\Gamma$，アルファベット $\Delta$ について $\Gamma$ から $\Delta^*$ への写像を**評価** ($evaluation$) という．変数の有限集合 $\Gamma$ とアルファベット $\Delta$ が与えられたとき，次の文法から生成される列すべてからなる集合を $E(\Gamma, \Delta)$ と表し，$\Gamma$ と $\Delta$ **上の表現の集合**と呼ぶ．

$$E ::= \varepsilon \mid aE \mid \gamma E$$

ここで $a \in \Delta$，$\gamma \in \Gamma$ である．$E(\Gamma, \Delta)$ は評価を更新するために使う表現の集合である．評価 $\alpha : \Gamma \to \Delta^*$ の自然な拡張として $E(\Gamma, \Delta)$ から $\Delta^*$ への写像が考えられる．表現 $e \in E(\Gamma, \Delta)$ について，$e$ に現れる変数 $\gamma$ を $\alpha(\gamma)$ で置換して得られる $\Delta$ 上の文字列を $e\alpha$ と表す．

$\Gamma$，$\Gamma'$ を変数の有限集合，$\Delta$ をアルファベットとしたとき，$\Gamma$ から $E(\Gamma', \Delta)$ への写像を**割当て** ($assignment$) と呼ぶ．**割当て** $\rho : \Gamma \to E(\Gamma', \Delta)$ を表すために次の表記を用いる．

$$\rho = [\gamma_1 := e_1, \ldots, \gamma_n := e_n]$$

ここで $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$，$e_i = \rho(\gamma_i) \in E(\Gamma', \Delta)$ である．この表記においてある変数 $\gamma$ に関する表現が明示的に書かれていないとき，$\gamma := \gamma$ であるものとする．また $\Gamma$，$\Gamma'$，$\Delta$ に対する割

当てすべてを含むの集合を $\mathcal{A}(\Gamma, \Gamma', \Delta)$ と表す．割当て $\rho : \Gamma \to E(\Gamma', \Delta)$ の自然な拡張として $E(\Gamma, \Delta)$ から $E(\Gamma', \Delta)$ への写像が考えられる．表現 $e \in E(\Gamma, \Delta)$ について，$e$ に現れる変数 $\gamma$ を $\rho(\gamma)$ で置換して得られる $\Gamma'$ と $\Delta$ 上の表現を $e\rho$ と表す．割当ての集合 $\mathcal{A}(\Gamma, \Gamma', \Delta)$ について $\Gamma' = \emptyset$ のとき，割当て $\alpha \in \mathcal{A}(\Gamma, \Gamma', \Delta)$ は $\Gamma$ から $\Delta^*$ への写像であり，これは評価に他ならない．そのため，割当てについて定義した表記を評価についても用いることとする．

二つの割当て $\rho_1 : \Gamma_1 \to E(\Gamma_1', \Delta)$，$\rho_2 : \Gamma_2 \to E(\Gamma_2', \Delta)$ が与えられたとき，$\Gamma_1$ から $\Gamma_1' \cup \Gamma_2'$ への割当て $\rho_1\rho_2$ を以下のように定義する．各 $\gamma \in \Gamma_1$ に対して，$\rho_1(\gamma) \in E(\Gamma_1', \Delta)$ に現れる $\gamma_1' \in \Gamma_1' \cap \Gamma_2$ を $\rho_2(\gamma_1') \in E(\Gamma_2', \Delta)$ で置換し，$\gamma_1' \in \Gamma_1' \setminus \Gamma_2$ はそのまま残して得られる $E(\Gamma_1' \cup \Gamma_2', \Delta)$ の元を $(\rho_1\rho_2)(\gamma)$ と定義する．二つの割当て $\rho_1 : \Gamma_1 \to E(\Gamma, \Delta)$，$\rho_2 : \Gamma_2 \to E(\Gamma, \Delta)$ について $\Gamma_1 \cap \Gamma_2 = \emptyset$ のとき， $\rho_1 \uplus \rho_2$ で次のように定義される $\Gamma_1 \uplus \Gamma_2$ から $E(\Gamma, \Delta)$ への写像 (割当て) を表す．

$$\rho_1 \uplus \rho_2 = [\gamma_1 := \rho_1(\gamma_1), \ldots, \gamma_n := \rho_1(\gamma_n), \gamma_1' := \rho_2(\gamma_1'), \ldots, \gamma_m' := \rho_2(\gamma_m')]$$

ただし $\Gamma_1 = \{\gamma_1, \ldots, \gamma_n\}$，$\Gamma_2 = \{\gamma_1', \ldots, \gamma_m'\}$ とする．

**例 5.** $\Gamma = \{\gamma_1, \gamma_2\}$，$\Delta = \{\mathtt{a}, \mathtt{b}\}$ とする．評価 $\alpha = [\gamma_1 := \mathtt{ab}, \gamma_2 := \mathtt{ba}] \in \mathcal{A}(\Gamma, \emptyset, \Delta)$，割当て $\rho_1 = [\gamma_1 := \gamma_1\gamma_2, \gamma_2 := \gamma_1\mathtt{b}] \in \mathcal{A}(\Gamma, \Gamma, \Delta)$ について，

$$\rho_1\alpha = \left[\gamma_1 := \underbrace{\mathtt{ab}}_{\gamma_1}\underbrace{\mathtt{ba}}_{\gamma_2}, \gamma_2 := \underbrace{\mathtt{ab}}_{\gamma_1}\mathtt{b}\right]$$

である．また，$\rho_2 = [\gamma_1 := \mathtt{a}\gamma_1\gamma_2\mathtt{b}, \gamma_2 := \gamma_2\gamma_1]$ としたとき，

$$\begin{aligned}\rho_2\rho_1\alpha &= \rho_2(\rho_1\alpha)\\ &= \rho_2\left[\gamma_1 := \mathtt{abba}, \gamma_2 := \mathtt{abb}\right]\\ &= \left[\gamma_1 := \mathtt{a}\underbrace{\mathtt{abba}}_{\gamma_1}\underbrace{\mathtt{abb}}_{\gamma_2}\mathtt{b}, \gamma_2 := \underbrace{\mathtt{abb}}_{\gamma_2}\underbrace{\mathtt{abba}}_{\gamma_1}\right]\end{aligned}$$

となる．また，次のように左側から先に計算しても結果は変わらない．

$$\begin{aligned}(\rho_2\rho_1)\alpha &= \left[\gamma_1 := \mathtt{a}\underbrace{\gamma_1\gamma_2}_{\gamma_1}\mathtt{b}\underbrace{\gamma_1\mathtt{b}}_{\gamma_2}, \gamma_2 := \underbrace{\gamma_1\mathtt{b}}_{\gamma_2}\underbrace{\gamma_1\gamma_2}_{\gamma_1}\right]\alpha\\ &= \left[\gamma_1 := \mathtt{a}\underbrace{\mathtt{ab}}_{\gamma_1}\underbrace{\mathtt{ba}}_{\gamma_2}\mathtt{b}\underbrace{\mathtt{ab}}_{\gamma_2}\mathtt{bb}, \gamma_1 := \underbrace{\mathtt{ab}}_{\gamma_1}\mathtt{b}\underbrace{\mathtt{ab}}_{\gamma_1}\underbrace{\mathtt{ba}}_{\gamma_2}\right]\end{aligned}$$

■

## 4.2 ランク付き木から文字列への決定性ストリーム変換器の定義

**定義 7.** ランク付き木から文字列への決定性ストリーム変換器 (*deterministic streaming ranked-tree-to-string transducer*, SRTST) は，次のように定義される組 $(S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ である．

- $S$ は**状態**の有限集合，
- $\Sigma$ は**入力**ランク付きアルファベット，
- $\Delta$ は**出力**アルファベット，
- $P$ は**スタック記号**の有限集合，
- $s_0 \in S$ は**初期状態**，
- $\Gamma$ は変数の有限集合，
- $F : S \rightharpoonup E(\Gamma, \Delta)$ は**出力関数**，
- $\delta_c : S \times \Sigma \to S \times P$ は**呼び出し遷移関数**，
- $\delta_r : S \times P \times \Sigma \to S$ は**戻り遷移関数**，
- $\rho_c : S \times \Sigma \to \mathcal{A}(\Gamma, \Gamma, \Delta)$ は**呼び出し更新関数**，
- $\rho_r : S \times P \times \Sigma \to \mathcal{A}(\Gamma, \Gamma \uplus \Gamma_p, \Delta)$ は**戻り更新関数**．変数集合 $\Gamma_p$ は $\Gamma \cap \Gamma_p = \emptyset$ であり，各変数 $\gamma \in \Gamma$ に対して変数 $\gamma_p \in \Gamma_p$ が一対一に対応するものとする． ■

呼び出し更新関数は，現在の状態と入力記号から $\Gamma$，$\Gamma$，$\Delta$ 上の割当てを返す．呼び出し更新関数により定まる割当ては，現在の評価から新たに評価を計算するために用いる．一方，戻り更新関数は，現在の状態，スタックに積まれたスタック記号，入力記号から $\Gamma$，$\Gamma \cup \Gamma_p$，$\Delta$ 上の割当てを返す．スタックのトップに積まれた評価を $\beta$ としたとき，戻り更新関数から定まる割当てに現れる $\gamma_p \in \Gamma_p$ は $\beta(\gamma)$ で置換される変数である．戻り遷移関数により定まる割当ては，現在の評価とスタックに積まれた評価から新たな評価を計算するために用いる．

$\Phi = S \times (P \times \mathcal{A}(\Gamma, \emptyset, \Delta))^* \times \mathcal{A}(\Gamma, \emptyset, \Delta)$ とする．$\Phi$ は SRTST の**様相** (*configuration*) の集合であり，$(s, \Lambda, \alpha) \in \Phi$ について，それぞれ $s$ は状態，$\Lambda$ はスタック，$\alpha$ は評価である．スタックにはスタック記号と評価の組が積まれる．$\alpha_\varepsilon = [\gamma := \varepsilon]_{\gamma \in \Gamma}$ とする．SRTST の遷移関数 $\delta : \Phi \times \hat{\Sigma} \rightharpoonup \Phi$ を定義する．初期様相を $(s_0, \varepsilon, \alpha_\varepsilon)$ とする．入力 $a \in \hat{\Sigma}$ が与えられたとき，様相上の遷移関数 $\delta(-, a)$ は次のように定義される．

**呼び出し遷移**　ある $b \in \Sigma$ について $a = \langle b$ のとき，様相 $(s, \Lambda, \alpha)$ に対し，$\delta((s, \Lambda, \alpha), a) = (s', (p, \alpha')\Lambda, \alpha_\varepsilon)$ とする．

- $s'$ と $p$ は $\delta_c(s, b) = (s', p)$ により得られる．
- $(p, \alpha')\Lambda$ は，スタック $\Lambda$ のトップに組 $(p, \alpha')$ をプッシュした後にできるスタックであるが，ここで $\alpha' = \rho_c(s, b)\alpha$ である．つまり，$\alpha'$ は $\rho_c(s, b)$ に現れる $\gamma \in \Gamma$ を

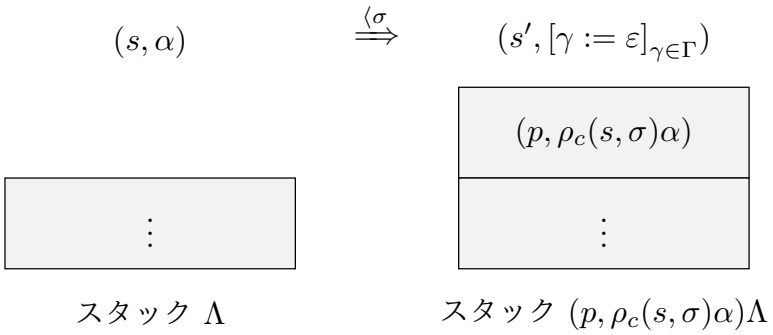


図 4.1　呼び出し遷移

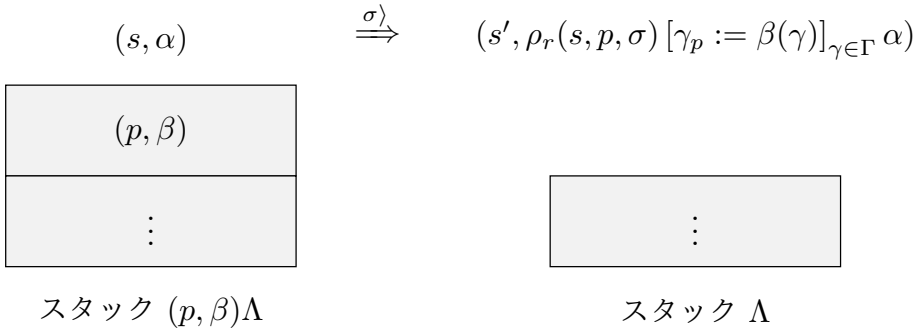


図 4.2　戻り遷移

$\alpha(\gamma)$ で置換して得られる評価である．

呼び出し遷移を図を用いて表すと図 4.1 のようになる．

**戻り遷移**　ある $b \in \Sigma$ について $a = b\rangle$ のとき，様相 $(s, (p, \beta)\Lambda, \alpha)$ に対し，$\delta((s, (p, \beta)\Lambda, \alpha), a) = (s', \Lambda, \alpha')$ とする．

- $s'$ は $\delta_r(s, p, b) = s'$ により得られる．
- $\Lambda$ は，スタック $(p, \beta)\Lambda$ のトップの組 $(p, \beta)$ をポップした後にできるスタックである．
- $\alpha' = \rho_r(s, p, b)\beta_p\alpha$ とする．ここで $\beta_p = [\gamma_p := \beta(\gamma)]_{\gamma\in\Gamma}$ とする．つまり，$\alpha'$ は $\rho_r(s, p, b)$ に現れる $\gamma_p \in \Gamma_p$ を $\beta(\gamma)$ で置換し，その結果得られた表現に現れる各 $\gamma \in \Gamma$ を $\alpha(\gamma)$ で置換して得られる評価である．

戻り遷移を図を用いて表すと図 4.2 のようになる．

SRTST $T$ について，入力 $a \in \hat{\Sigma}$ による様相 $c$ から $c'$ への遷移を $c \overset{a}{\Rightarrow}_T c'$ と表し，$c' = \delta(c, a)$ で定義する．$T$ が明らかなとき単に $c \overset{a}{\Rightarrow} c'$ と書く．様相 $c$ から 0 回以上の遷移で様相 $c'$ へ至るとき $c \Rightarrow^* c'$ と書く．遷移関数 $\delta$ の $\hat{\Sigma}^*$ 上への自然な拡張を $\hat{\delta}$ と表す．ネスト文字列 $w \in \hat{\Sigma}^*$ による様相 $c$ から $c'$ への遷移を $c \overset{w}{\Rightarrow}_T c'$ と表し，$c' = \hat{\delta}(c, w)$ で定義する．$T$ に対して部分関数 $[\![T]\!] : \lfloor\mathcal{T}_\Sigma\rfloor \rightharpoonup \Delta^*$ を次のように定義する．ランク付きネスト文字列 $w \in \lfloor\mathcal{T}_\Sigma\rfloor$ について

$\hat{\delta}((s_0, \varepsilon, \alpha_\varepsilon), w) = (s, \varepsilon, \alpha)$ のとき，$F(s)$ が未定義ならば $[\![T]\!](w)$ も未定義とし，そうでないならば $[\![T]\!](w) = F(s)\alpha$ とする．$[\![T]\!]$ の定義域を $\mathrm{Dom}(T)$ で表す．

**例 6.** 例 4 で示した $\mathrm{yDT}^R$ と等価な変換を行う SRTST を定義する．$T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ を SRTST とする．ここで $S = \{s_?, s_\mathtt{a}, s_\mathtt{b}\}$，$\Sigma = \{\mathtt{f}^{(2)}, \mathtt{a}^{(0)}, \mathtt{b}^{(0)}\}$，$\Delta = \{\mathtt{a}, \mathtt{b}\}$，$P = \{p_?, p_\mathtt{a}, p_\mathtt{b}\}$，$s_0 = s_?$，$\Gamma = \{\gamma\}$ とする．遷移関数 $\delta_c$ をすべての $\sigma \in \Sigma$，$h \in \{\mathtt{a}, \mathtt{b}, ?\}$ について，$\delta_c(s_h, \sigma) = (s_?, p_h)$ とする．遷移関数 $\delta_r$ をすべての $s \in S$，$\sigma \in \Sigma$ について，

$$\begin{aligned}
&\delta_r(s, p_\mathtt{a}, \sigma) = s_\mathtt{a}, && \delta_r(s_\mathtt{a}, p_?, \sigma) = s_\mathtt{a}, && \delta_r(s_?, p_?, \mathtt{a}) = s_\mathtt{a},\\
&\delta_r(s, p_\mathtt{b}, \sigma) = s_\mathtt{b}, && \delta_r(s_\mathtt{b}, p_?, \sigma) = s_\mathtt{b}, && \delta_r(s_?, p_?, \mathtt{b}) = s_\mathtt{b}.
\end{aligned}$$

とする．呼び出し更新関数 $\rho_c$ をすべての $s \in S$ について，

$$\begin{aligned}
&\rho_c(s_?, \mathtt{a}) = [\gamma := \mathtt{a}], \quad \rho_c(s_\mathtt{a}, \mathtt{a}) = [\gamma := \gamma\mathtt{a}], \quad \rho_c(s_\mathtt{a}, \mathtt{b}) = [\gamma := \gamma\mathtt{b}], \quad \rho_c(s, \mathtt{f}) = [\gamma := \gamma],\\
&\rho_c(s_?, \mathtt{b}) = [\gamma := \mathtt{b}], \quad \rho_c(s_\mathtt{b}, \mathtt{a}) = [\gamma := \mathtt{a}\gamma], \quad \rho_c(s_\mathtt{b}, \mathtt{b}) = [\gamma := \mathtt{b}\gamma]
\end{aligned}$$

とする．戻り更新関数 $\rho_r$ をすべての $s \in S$ について，

$$\begin{aligned}
&\rho_r(s, p_?, \mathtt{a}) = [\gamma := \gamma_p], && \rho_r(s, p_\mathtt{a}, \mathtt{f}) = [\gamma := \gamma_p\gamma], && \rho_r(s, p_?, \mathtt{f}) = [\gamma := \gamma],\\
&\rho_r(s, p_?, \mathtt{b}) = [\gamma := \gamma_p], && \rho_r(s, p_\mathtt{b}, \mathtt{f}) = [\gamma := \gamma\gamma_p].
\end{aligned}$$

とする．出力関数を，すべての $s \in S$ について $F(s) = \gamma$ とする．入力として `f(f(b,a),f(a,b)))` を $T$ に与えたとき，遷移は次のようになる．ここでは見易さのためにスタックを $[\![-]\!]$ で囲っている．

$$
\begin{aligned}
& (s_?, [\![]\!] && , \alpha_\varepsilon && ) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} & (s_?, [\![(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} & (s_?, [\![(p_?, \alpha_\varepsilon)(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\langle \mathtt{b}}{\Longrightarrow} & (s_?, [\![(p_?, [\gamma := \mathtt{b}])(p_?, \alpha_\varepsilon)(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\mathtt{b}\rangle}{\Longrightarrow} & (s_\mathtt{b}, [\![(p_?, \alpha_\varepsilon)(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{b}] && ) \\
\overset{\langle \mathtt{a}}{\Longrightarrow} & (s_?, [\![(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\mathtt{a}\rangle}{\Longrightarrow} & (s_\mathtt{b}, [\![(p_?, \alpha_\varepsilon)(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{ab}] && ) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} & (s_\mathtt{b}, [\![(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{ab}] && ) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} & (s_?, [\![(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\langle \mathtt{a}}{\Longrightarrow} & (s_?, [\![(p_?, [\gamma := \mathtt{a}])(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\mathtt{a}\rangle}{\Longrightarrow} & (s_\mathtt{a}, [\![(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{a}] && ) \\
\overset{\langle \mathtt{b}}{\Longrightarrow} & (s_?, [\![(p_\mathtt{a}, [\gamma := \mathtt{ab}])(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)]\!] && , \alpha_\varepsilon && ) \\
\overset{\mathtt{b}\rangle}{\Longrightarrow} & (s_\mathtt{a}, [\![(p_\mathtt{b}, [\gamma := \mathtt{ab}])(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{ab}] && ) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} & (s_\mathtt{b}, [\![(p_?, \alpha_\varepsilon)]\!] && , [\gamma := \mathtt{abab}]) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} & (s_\mathtt{b}, [\![]\!] && , [\gamma := \mathtt{abab}])
\end{aligned}
\tag{4.1}
\tag{4.2}
$$

最後に $F(s_\mathtt{b}) = \gamma$ から $[\![T]\!]\,(\mathtt{f(f(b,a),f(a,b))}) = \mathtt{abab}$ となる．式 (4.1) において，呼び出し記号 $\langle\mathtt{a}$ を読んだときの変数とスタックの様子を図 4.3 に示す．また，式 (4.2) において，戻り記号 $\mathtt{f}\rangle$ を読んだときの変数とスタックの様子を図 4.4 に示す．図 4.3 と図 4.4 の $\overset{a/\rho}{\Longrightarrow}$ において $a \in \hat{\Sigma}$ であり，$\rho$ は更新関数により定まる割当てである． ■

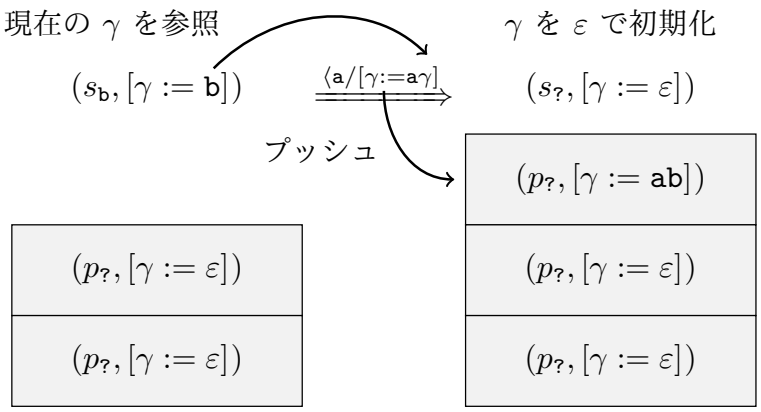


図 4.3　呼び出し遷移 (式 (4.1))

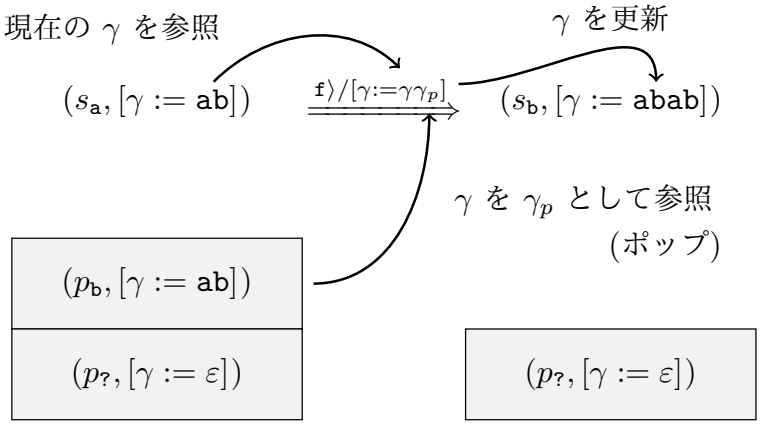


図 4.4　戻り遷移 (式 (4.2))

# 5　SRTST と $\mathrm{yDT}^R$ の表現力の同等性

本章では，5.1 節で任意の SRTST から等価な変換を定義する $\mathrm{yDT}^R$，5.3 節で任意の $\mathrm{yDT}^R$ から等価な変換を定義する SRTST が構築可能であることを示す．5.5 節で SRTST の表現力について述べる．

## 5.1　$\mathrm{yDT}^R$ から SRTST の構成

本節では，任意の $\mathrm{yDT}^R$ から等価な SRTST を構成する方法を示す．ここで構成する SRTST は**上昇型** (*bottom-up*) である．SRTST が上昇型であるとは，呼び出し遷移関数は常に初期状態を返し，呼び出し更新関数が常に現在の評価を変えないを割当て返す SRTST のことである．つまり，すべての $s \in S$，$\sigma \in \Sigma$ について $\delta_c(s,\sigma) = (s', p)$ のとき，$s' = s_0$ かつ $\rho_c(s,\sigma) = [\gamma := \gamma]_{\gamma\in\Gamma}$ であるような SRTST のことをいう．

**補題 1.** $M$ を任意の $\mathrm{yDT}^R$ とするとき，すべての $t \in \mathrm{Dom}(M)$ について $[\![M]\!](t) = [\![T]\!](\lfloor t \rfloor)$ を満たす SRTST $T$ が構築可能である．■

**証明.** $M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ を $\mathrm{yDT}^R$ とする．ただし，$(\Sigma, \Pi, \theta)$ は DBTA とする．すべての $t \in \mathrm{Dom}(M)$ について $[\![M]\!](t) = [\![T]\!](\lfloor t \rfloor)$ を満たす SRTST $T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ を構築する．$m = \max(\{1\} \cup \{\mathrm{rank}(\sigma) \mid \sigma \in \Sigma\})$ とし，$S = \Pi^{\leq m}$，$\Gamma = Q \times X_m$，$P = \Pi^{\leq m}$，$s_0 = ()$ とする．呼び出し遷移関数 $\delta_c$ を次のように定義する．すべての $\sigma^{(n)} \in \Sigma$，$\boldsymbol{\pi} \in S$ について，

$$\delta_c(\boldsymbol{\pi}, \sigma) = ((), \boldsymbol{\pi})$$

とする．戻り遷移関数 $\delta_r$ を次のように定義する．すべての $\sigma^{(n)} \in \Sigma$，$\boldsymbol{\pi} \in \Pi^n$，$\boldsymbol{\pi}' \in \Pi^{\leq m-1}$ について，

$$\delta_r(\boldsymbol{\pi}, \boldsymbol{\pi}', \sigma) = \boldsymbol{\pi}' \mid\mid \theta(\sigma, \boldsymbol{\pi})$$

とする．次に呼び出し更新関数 $\rho_c$ を次のように定義する．すべての $\boldsymbol{\pi} \in S$，$\sigma \in \Sigma$ について，

$$\rho_c(\boldsymbol{\pi}, \sigma) = [\gamma := \gamma]_{\gamma\in\Gamma}$$

とする．戻り更新関数 $\rho_r$ を次のように定義する．すべての $\sigma^{(n)} \in \Sigma$，$\boldsymbol{\pi} \in \Pi^n$，$\boldsymbol{\pi}' \in \Pi^{\leq m-1}$ について，

$$\rho_r(\boldsymbol{\pi}, \boldsymbol{\pi}', \sigma) = \left[q(x_{|\boldsymbol{\pi}'|+1}) := \mathcal{U}(q, \sigma, \boldsymbol{\pi})\right]_{q\in Q} \uplus \left[q(x_k) := q(x_k)_p\right]_{q\in Q,\, x_k \in X_{|\boldsymbol{\pi}'|}}$$

写像 $\mathcal{U} : Q \times \Sigma^{(n)} \times \Pi^n \to E(\Gamma, \Delta)$ を次のように定義する．

$$\mathcal{U}(q, \sigma, \boldsymbol{\pi}) = \begin{cases} \mathrm{rhs}(q, \sigma, \boldsymbol{\pi}) & (q, \sigma, \boldsymbol{\pi})\text{-規則が存在する場合} \\ \varepsilon & \text{それ以外の場合} \end{cases}$$

また，$\mathcal{U}$ の定義からすべての $x_l \in X_m \setminus X_{|\boldsymbol{\pi}'|+1}$ について $q(x_l)$ が変数の更新に使われることはないため $[q(x_l) := q(x_l)]_{q \in Q,\, x_l \in X_m \setminus X_{|\boldsymbol{\pi}'|+1}}$ のようにすればよい．最後に出力関数 $F$ をすべての $(\tau, \pi) \in \mathit{Init}$ について $F((\pi)) = \tau$ とする．

ここで示した構成では $\rho_r$ に関して $(q, \sigma, \boldsymbol{\pi})$-規則が存在しないとき，変数 $q(x_{|\boldsymbol{\pi}'|+1})$ を $\varepsilon$ で置換することにしている．本論文では $\mathrm{yDT}^R$ が式 (3.1) を満たすことを仮定しているため，ある入力木 $t \in \mathcal{T}_\Sigma$ について $(q, \sigma, \boldsymbol{\pi})$-規則が存在しないとき，$t \notin \mathrm{Dom}(M)$ であるため $t \notin \bigcup_{(\tau,\pi) \in \mathit{Init}} \mathcal{L}(\pi)$ となる．このことから $(q, \sigma, \boldsymbol{\pi})$-規則が存在しないとき，$t \in \mathcal{L}(\pi')$ を満たす $\pi'$ について $F$ が定義されないことは構成法から明らかである．よって $\lfloor t \rfloor$ は $[\![\tau]\!]_M$ の定義域に含まれないため，$q(x_{|\boldsymbol{\pi}'|+1})$ の値を $\varepsilon$ に更新してもよい． □

これにより，任意の $\mathrm{yDT}^R$ と等価な変換を行う SRTST を構成することができる．また，構成される SRTST は明らかに上昇型である．

この構成法から作られる SRTST の例を示す．

**例 7.** 例 4 の $\mathrm{yDT}^R$ $M$ と等価な変換を行う SRTST $T$ を補題 1 で示した方法により構築する．$T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ とし，状態の有限集合 $S$，スタック記号の有限集合 $P$，変数の有限集合 $\Gamma$ は，それぞれ以下のように定義される．

$$\begin{aligned} S &= \{(), (\pi_\mathtt{a}), (\pi_\mathtt{b}), (\pi_\mathtt{a}, \pi_\mathtt{a}), (\pi_\mathtt{a}, \pi_\mathtt{b}), (\pi_\mathtt{b}, \pi_\mathtt{b}), (\pi_\mathtt{b}, \pi_\mathtt{a})\} && = \Pi^{\leq 2}, \\ P &= \{(), (\pi_\mathtt{a}), (\pi_\mathtt{b}), (\pi_\mathtt{a}, \pi_\mathtt{a}), (\pi_\mathtt{a}, \pi_\mathtt{b}), (\pi_\mathtt{b}, \pi_\mathtt{b}), (\pi_\mathtt{b}, \pi_\mathtt{a})\} && = \Pi^{\leq 2}, \\ \Gamma &= \{q(x_1), q(x_2)\} && = Q \times X_2. \end{aligned}$$

初期状態は $s_0 = ()$ である．呼び出し遷移関数 $\delta_c$ は，すべての $\sigma \in \Sigma$ について次のように定義される．

$$\delta_c((), \sigma) = ((), ()), \qquad \delta_c((\pi_\mathtt{a}), \sigma) = ((), (\pi_\mathtt{a})), \qquad \delta_c((\pi_\mathtt{b}), \sigma) = ((), (\pi_\mathtt{b})).$$

ここですべての $\boldsymbol{\pi} \in \Pi^2$ について定義を省略しているが，入力されるネスト文字列を $\lfloor \mathcal{T}_\Sigma \rfloor$ の要素に限っているため，状態 $\boldsymbol{\pi}$ のとき呼び出し記号を読むことがないからである．戻り遷移関数 $\delta_r$ は，すべての $\pi \in \Pi$ について次のように定義される．

$$\begin{aligned} &\delta_r((), () \quad, \mathtt{a}) = (\pi_\mathtt{a}), && \delta_r((\pi_\mathtt{a}, \pi), () \quad, \mathtt{f}) = (\pi_\mathtt{a}), \\ &\delta_r((), () \quad, \mathtt{b}) = (\pi_\mathtt{b}), && \delta_r((\pi_\mathtt{b}, \pi), () \quad, \mathtt{f}) = (\pi_\mathtt{b}), \\ &\delta_r((), (\pi_\mathtt{a}), \mathtt{a}) = (\pi_\mathtt{a}, \pi_\mathtt{a}), && \delta_r((\pi_\mathtt{a}, \pi), (\pi_\mathtt{a}), \mathtt{f}) = (\pi_\mathtt{a}, \pi_\mathtt{a}), \\ &\delta_r((), (\pi_\mathtt{a}), \mathtt{b}) = (\pi_\mathtt{a}, \pi_\mathtt{b}), && \delta_r((\pi_\mathtt{b}, \pi), (\pi_\mathtt{a}), \mathtt{f}) = (\pi_\mathtt{a}, \pi_\mathtt{b}), \\ &\delta_r((), (\pi_\mathtt{b}), \mathtt{a}) = (\pi_\mathtt{b}, \pi_\mathtt{a}), && \delta_r((\pi_\mathtt{a}, \pi), (\pi_\mathtt{b}), \mathtt{f}) = (\pi_\mathtt{b}, \pi_\mathtt{a}), \\ &\delta_r((), (\pi_\mathtt{b}), \mathtt{b}) = (\pi_\mathtt{b}, \pi_\mathtt{b}), && \delta_r((\pi_\mathtt{b}, \pi), (\pi_\mathtt{b}), \mathtt{f}) = (\pi_\mathtt{b}, \pi_\mathtt{b}) \end{aligned}$$

$\delta_r$ についても全域的な定義となっていないが，定義されていない域に関して呼び出されることがないため上記の定義で十分である．呼び出し更新関数 $\rho_c$ は，すべての $s \in S$，$\sigma \in \Sigma$ について次のように定義される．

$$\rho_c(s, \sigma) = [\gamma := \gamma]_{\gamma \in \Gamma}$$

戻り更新関数 $\rho_r$ は，すべての $\pi_1, \pi_2 \in \Pi$ について次のように定義される．

$$\begin{aligned}
&\rho_r((), (), \mathtt{a}) = [q(x_1) := \mathtt{a}]\,, \\
&\rho_r((), (), \mathtt{b}) = [q(x_1) := \mathtt{b}]\,, \\
&\rho_r((), (\pi_1), \mathtt{a}) = [q(x_2) := \mathtt{a}, q(x_1) := q(x_1)_p]\,, \\
&\rho_r((), (\pi_1), \mathtt{b}) = [q(x_2) := \mathtt{b}, q(x_1) := q(x_1)_p]\,, \\
&\rho_r((\pi_\mathtt{a}, \pi_1), (), \mathtt{f}) = [q(x_1) := q(x_1)q(x_2)]\,, \\
&\rho_r((\pi_\mathtt{b}, \pi_1), (), \mathtt{f}) = [q(x_1) := q(x_2)q(x_1)]\,, \\
&\rho_r((\pi_\mathtt{a}, \pi_1), (\pi_2), \mathtt{f}) = [q(x_2) := q(x_1)q(x_2), q(x_1) := q(x_1)_p]\,, \\
&\rho_r((\pi_\mathtt{b}, \pi_1), (\pi_2), \mathtt{f}) = [q(x_2) := q(x_2)q(x_1), q(x_1) := q(x_1)_p]\,.
\end{aligned}$$

出力関数は $F((\pi_\mathtt{a})) = q(x_1)$，$F((\pi_\mathtt{b})) = q(x_1)$ となる．

ここで構成した $T$ に対して `f(f(b,a),f(a,b))` を入力したときの遷移は次のようになる．

$$\begin{aligned}
&((), \llbracket\rrbracket, \alpha_\varepsilon) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} &((), \llbracket((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} &((), \llbracket((), \alpha_\varepsilon)((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\langle \mathtt{b}}{\Longrightarrow} &((), \llbracket((), \alpha_\varepsilon)((), \alpha_\varepsilon)((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\mathtt{b}\rangle}{\Longrightarrow} &((\pi_\mathtt{b}), \llbracket((), \alpha_\varepsilon)((), \alpha_\varepsilon)\rrbracket, [q(x_1) := \mathtt{b}]) \\
\overset{\langle \mathtt{a}}{\Longrightarrow} &((), \llbracket((\pi_\mathtt{b}), [q(x_1) := \mathtt{b}])((), \alpha_\varepsilon)((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\mathtt{a}\rangle}{\Longrightarrow} &((\pi_\mathtt{b}, \pi_\mathtt{a}), \llbracket((), \alpha_\varepsilon)((), \alpha_\varepsilon)\rrbracket, [q(x_2) := \mathtt{a}, q(x_1) := \mathtt{b}]) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} &((\pi_\mathtt{b}), \llbracket((), \alpha_\varepsilon)\rrbracket, [q(x_1) := \mathtt{ab}]) \\
\overset{\langle \mathtt{f}}{\Longrightarrow} &((), \llbracket((\pi_\mathtt{b}), [q(x_1) := \mathtt{ab}])((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\langle \mathtt{a}}{\Longrightarrow} &((), \llbracket((), \alpha_\varepsilon)((\pi_\mathtt{b}), [q(x_1) := \mathtt{ab}])((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\mathtt{a}\rangle}{\Longrightarrow} &((\pi_\mathtt{a}), \llbracket((\pi_\mathtt{b}), [q(x_1) := \mathtt{ab}])((), \alpha_\varepsilon)\rrbracket, [q(x_1) := \mathtt{a}]) \\
\overset{\langle \mathtt{b}}{\Longrightarrow} &((), \llbracket((\pi_\mathtt{a}), [q(x_1) := \mathtt{a}])((\pi_\mathtt{b}), [q(x_1) := \mathtt{ab}])((), \alpha_\varepsilon)\rrbracket, \alpha_\varepsilon) \\
\overset{\mathtt{b}\rangle}{\Longrightarrow} &((\pi_\mathtt{a}, \pi_\mathtt{b}), \llbracket((\pi_\mathtt{b}), [q(x_1) := \mathtt{ab}])((), \alpha_\varepsilon)\rrbracket, [q(x_2) := \mathtt{b}, q(x_1) := \mathtt{a}]) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} &((\pi_\mathtt{b}, \pi_\mathtt{a}), \llbracket((), \alpha_\varepsilon)\rrbracket, [q(x_2) := \mathtt{ab}, q(x_1) := \mathtt{ab}]) \\
\overset{\mathtt{f}\rangle}{\Longrightarrow} &((\pi_\mathtt{b}), \llbracket\rrbracket, [q(x_1) := \mathtt{abab}])
\end{aligned}$$

出力関数が $F(\pi_{\mathtt{b}}) = q(x_1)$ であるから $[\![T]\!]\,(\mathtt{f(f(b,a),f(a,b))}) = \mathtt{abab}$ となる． ■

## 5.2 $\mathrm{yDT}^R$ からの SRTST の構成法の正当性

本節では，5.1 節 で示した構成法が正しいことを示す．はじめに構成された SRTST の定義域と元となった $\mathrm{yDT}^R$ の定義域が等しいことを示し，その後，$\mathrm{yDT}^R$ の定義域において等価な変換となっていることを示す．

$\mathrm{yDT}^R$ $M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ とし，補題 1 の証明で示した方法により構成される SRTST を $T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ とする．

**補題 2.** すべての $t \in \mathcal{T}_\Sigma$，$\boldsymbol{\pi} \in \Pi$ について $\delta((\boldsymbol{\pi}, \Lambda, \alpha), \lfloor t \rfloor) = (\boldsymbol{\pi} \,||\, \hat{\theta}(t), \Lambda, \alpha')$ である． ■

**証明.** $t$ 上の帰納法により証明する．$t_1, \ldots, t_n \in \mathcal{T}_\Sigma$ について $\pi_1 = \hat{\theta}(t_1), \ldots, \pi_n = \hat{\theta}(t_n)$ とし，各 $i \in [n]$ において $\delta((\boldsymbol{\pi_i}, \Lambda_i, \alpha_i), \lfloor t_i \rfloor) = (\boldsymbol{\pi_i} \,||\, \hat{\theta}(t_i), \Lambda_i, \alpha_i')$ が成り立つと仮定する．$t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$ について，遷移 $\delta((\boldsymbol{\pi}, \Lambda, \alpha), \lfloor t \rfloor)$ は帰納法の仮定より次のようになる．

$$
\begin{aligned}
(\boldsymbol{\pi}, \Lambda, \alpha) &\xRightarrow{\langle\sigma} ((), (\boldsymbol{\pi}, \alpha)\Lambda, \alpha_0) \\
&\xRightarrow{\lfloor t_1 \rfloor} ((\pi_1), (\boldsymbol{\pi}, \alpha)\Lambda, \alpha_1) \\
&\Longrightarrow^* \cdots \\
&\xRightarrow{\lfloor t_n \rfloor} ((\pi_1, \ldots, \pi_n), (\boldsymbol{\pi}, \alpha)\Lambda, \alpha_n) \\
&\xRightarrow{\sigma\rangle} (\boldsymbol{\pi} \,||\, \theta(\sigma, \pi_1, \ldots, \pi_n), \Lambda, \alpha')
\end{aligned}
$$

ここで $(\Sigma, \Pi, \theta)$ が DBTA であることから $\hat{\theta}(\sigma(t_1, \ldots, t_n)) = \theta(\sigma, \hat{\theta}(t_1), \ldots, \hat{\theta}(t_n)) = \theta(\sigma, \pi_1, \ldots, \pi_n)$ である．よって，$\delta((\boldsymbol{\pi}, \Lambda, \alpha), \lfloor t \rfloor) = (\boldsymbol{\pi} \,||\, \hat{\theta}(t), \Lambda, \alpha')$ が成り立つ． □

**補題 3.** $T$ と $M$ の定義域は等しい． ■

**証明.** 式 (3.1) から $\mathrm{Dom}(M) = \biguplus_{(\tau,\pi) \in \mathit{Init}} \mathcal{L}(\pi)$，補題 2 から，すべての $t \in \mathcal{T}_\Sigma$ について $\delta(((), \varepsilon, \alpha_0), \lfloor t \rfloor) = ((\hat{\theta}(t)), \varepsilon, \alpha)$ である．よって，補題 1 の $\mathit{Init}$ の構成から

$$
\begin{aligned}
\lfloor t \rfloor \in \mathrm{Dom}(T) &\iff (\hat{\theta}(t)) \in \mathrm{Dom}(F) \\
&\iff (\tau, \hat{\theta}(t)) \in \mathit{Init} \\
&\iff t \in \mathrm{Dom}(M)
\end{aligned}
$$

である． □

**補題 4.** $\mathrm{yDT}^R$ $M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ とする．任意の $\tau \in \mathrm{Rhs}_{Q,\Delta}(X_n)$，$t_1 \ldots, t_n \in \mathcal{T}_\Sigma$ について，すべての $q(x_i) \in \tau$ において $t_i \in \mathrm{Dom}(q)$ のとき，

$$
[\![\tau]\!]_M\,(t_1, \ldots, t_n) = \tau\,[q(x_i) := [\![q]\!]_M\,(t_i)]_{q(x_i) \in \tau}
$$

が成り立つ．■

**証明**. $t$ と $\tau$ 上の帰納法による．□

**補題 5.** $m = \max(\{1\} \cup \{\mathrm{rank}(\sigma) \mid \sigma \in \Sigma\})$ とする．すべての $t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$, $\boldsymbol{\pi} \in \Pi^{\leq m-1}$ について $\delta((\boldsymbol{\pi}, \Lambda, \alpha), \lfloor t \rfloor) = (\boldsymbol{\pi}', \Lambda, \alpha')$ のとき，

$$\alpha' = \left[q(x_{|\boldsymbol{\pi}|+1}) := [\![q]\!]_M(t)\right]_{q \in Q} \uplus \left[q(x_i) := \alpha(q(x_i))\right]_{q \in Q,\, x_i \in X_{|\boldsymbol{\pi}|}}$$

である．ただし，$[\![q]\!]_M(t)$ が未定義のとき，ここでは $[\![q]\!]_M(t) = \varepsilon$ とする．■

**証明**. $t$ 上の帰納法により証明する．$t_1, \ldots, t_n \in \mathcal{T}_\Sigma$ について，各 $i \in [n]$ において $\delta((\boldsymbol{\pi_i}, \Lambda_i, \alpha_i), \lfloor t_i \rfloor) = (\boldsymbol{\pi_i'}, \Lambda_i, \alpha_i')$ のとき，

$$\alpha' = \left[q(x_{|\boldsymbol{\pi_i}|+1}) := [\![q]\!]_M(t_i)\right]_{q \in Q} \uplus \left[q(x_j) := \alpha_i(q(x_j))\right]_{q \in Q, x_j \in X_{|\boldsymbol{\pi_i}|}}$$

が成り立っていると仮定する．すべての $\sigma^{(n)} \in \Sigma$ について，$\delta((\boldsymbol{\pi}, \Lambda, \alpha), \lfloor \sigma(t_1, \ldots, t_n) \rfloor) = (\boldsymbol{\pi}', \Lambda, \alpha')$ における各遷移は，帰納法の仮定と $T$ の構成から次のようになる．

$$\begin{aligned}
&(\boldsymbol{\pi}, \Lambda, \alpha) \\
&\xrightarrow{\langle\sigma} ((), (\boldsymbol{\pi}, \rho_c(\boldsymbol{\pi}, \sigma)\alpha)\Lambda, \alpha_0) \\
&\xrightarrow{\lfloor t_1 \rfloor} ((\pi_1''), (\boldsymbol{\pi}, \rho_c(\boldsymbol{\pi}, \sigma)\alpha)\Lambda, \left[q(x_1) := [\![q]\!]_M(t_1)\right]_{q \in Q} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_1}) \\
&\Longrightarrow^* \cdots \\
&\xrightarrow{\lfloor t_n \rfloor} (\boldsymbol{\pi}'', (\boldsymbol{\pi}, \rho_c(\boldsymbol{\pi}, \sigma)\alpha)\Lambda, \left[q(x_i) := [\![q]\!]_M(t_i)\right]_{q \in Q,\, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n}) \\
&\xrightarrow{\sigma\rangle} (\boldsymbol{\pi}', \Lambda, \rho_r(\boldsymbol{\pi}'', \boldsymbol{\pi}, \sigma)\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q \in Q,\, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n}))
\end{aligned}$$

ただし，$\boldsymbol{\pi}'' = (\pi_1'', \ldots, \pi_n'')$，$\beta = [\gamma_p := \gamma\rho_c(\boldsymbol{\pi}, \sigma)\alpha]_{\gamma \in \Gamma} = [\gamma_p := \alpha(\gamma)]_{\gamma \in \Gamma}$ とする．はじめに $\alpha'(q(x_{|\boldsymbol{\pi}|+1}))$ を計算すると次のようになる．

$$\begin{aligned}
&\alpha'(q(x_{|\boldsymbol{\pi}|+1})) \\
&= q(x_{|\boldsymbol{\pi}|+1})\rho_r(\boldsymbol{\pi}'', \boldsymbol{\pi}, \sigma)\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q \in Q,\, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n}) \\
&= q(x_{|\boldsymbol{\pi}|+1})(\left[q(x_{|\boldsymbol{\pi}|+1}) := \mathcal{U}(q, \sigma, \boldsymbol{\pi}'')\right]_{q \in Q} \uplus \left[q(x_k) := q(x_k)_p\right]_{q \in Q,\, x_k \in X_{|\boldsymbol{\pi}'|}}) \\
&\qquad \beta([q(x_i) := [\![q]\!]_M(t_i)]_{q \in Q, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n}) \\
&= \mathcal{U}(q, \sigma, \boldsymbol{\pi}'')\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q \in Q,\, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n})
\end{aligned}$$

ここで $\sigma(t_1, \ldots, t_n) \notin \mathrm{Dom}(q)$ のとき，

$$\begin{aligned}
\alpha'(q(x_{|\boldsymbol{\pi}|+1})) &= \varepsilon\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q \in Q,\, x_i \in X_n} \uplus \left[q(x) := \varepsilon\right]_{q \in Q,\, x \in X_m \setminus X_n}) \\
&= \varepsilon
\end{aligned}$$

となる．一方，$t \in \mathrm{Dom}(q)$ のとき，ある $(q, \sigma, \boldsymbol{\pi}'')$-規則が存在して，

$$\begin{aligned}
&\alpha'(q(x_{|\boldsymbol{\pi}|+1})) \\
&= \mathrm{rhs}(q, \sigma, \boldsymbol{\pi}'')\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= \mathrm{rhs}(q, \sigma, \boldsymbol{\pi}'')\,[\gamma_p := \alpha(\gamma)]_{\gamma\in\Gamma}\,([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= \mathrm{rhs}(q, \sigma, \boldsymbol{\pi}'')\,[q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n}
\end{aligned}$$

となる．補題 2 から $t_1 \in \mathcal{L}(\pi_1''), \ldots, t_n \in \mathcal{L}(\pi_n'')$ であることと補題 4 より，

$$\begin{aligned}
\alpha'(q(x_{|\boldsymbol{\pi}|+1})) &= \mathrm{rhs}(q, \sigma, \boldsymbol{\pi}'')\,[q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \\
&= [\![\mathrm{rhs}(q, \sigma, \boldsymbol{\pi}'')]\!]_M(t_1, \ldots, t_n) \\
&= [\![q]\!]_M(\sigma(t_1, \ldots, t_n))
\end{aligned}$$

が成り立つ．

次にすべての $q \in Q$，$x_j \in X_{|\boldsymbol{\pi}|}$ について $\alpha'(q(x_j)) = \alpha(q(x_j))$ が成り立つこと示す．$\alpha'(q(x_j))$ を計算すると，

$$\begin{aligned}
\alpha'(q(x_j)) &= q(x_j)([q(x_{|\boldsymbol{\pi}|+1}) := \mathcal{U}(q, \sigma, \boldsymbol{\pi}'')]_{q\in Q} \uplus [q(x_k) := q(x_k)_p]_{q\in Q,\, x\in X_{|\boldsymbol{\pi}'|}}) \\
&\qquad \beta([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= q(x_j)_p\beta([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= q(x_j)_p\,[\gamma_p := \alpha(\gamma)]_{\gamma\in\Gamma}\,([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= \alpha(q(x_j))([q(x_i) := [\![q]\!]_M(t_i)]_{q\in Q,\, x_i \in X_n} \uplus [q(x) := \varepsilon]_{q\in Q,\, x\in X_m \setminus X_n}) \\
&= \alpha(q(x_j))
\end{aligned}$$

となる．よって，$\alpha' = [q(x_{|\boldsymbol{\pi}|+1}) := [\![q]\!]_M(t)]_{q\in Q} \uplus [q(x_i) := \alpha(q(x_i))]_{q\in Q, x_i \in X_{|\boldsymbol{\pi}|}}$ である． □

**補題 6.** すべての $t \in \mathrm{Dom}(M)$ について $[\![M]\!](t) = [\![T]\!](\lfloor t \rfloor)$ である． ■

**証明.** 補題 3 より $\mathrm{Dom}(M) = \mathrm{Dom}(T)$ である．また，補題 5 よりすべての $t \in \mathcal{T}_\Sigma$ について $\delta(((), \varepsilon, \alpha_0)) = ((\hat{\theta}(t)), \varepsilon, \alpha)$ のとき $\alpha = [q(x_1) := [\![q]\!]_M(t)]_{q\in Q}$ である．よって，

$$\begin{aligned}
[\![T]\!](\lfloor t \rfloor) &= F((\hat{\theta}(t)))\alpha \\
&= \mathit{Init}(\hat{\theta}(t))\,[q(x_1) := [\![q]\!]_M(t)]_{q\in Q} \\
&= \left[\!\!\left[\mathit{Init}(\hat{\theta}(t))\right]\!\!\right]_M(t) \\
&= [\![M]\!](t)
\end{aligned}$$

となる． □

## 5.3 SRTST から yDT$^R$ の構成

本節では，任意の SRTST から等価な yDT$^R$ を構成する方法を示す．

**補題 7.** $T$ を任意の SRTST とするとき，すべての $w \in \mathrm{Dom}(T)$ について $[\![T]\!](w) = [\![M]\!](\lceil w \rceil)$ を満たす yDT$^R$ $M$ が構築可能である． ■

**証明.** $T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ を SRTST とする．すべての $w \in \mathrm{Dom}(T)$ について $[\![T]\!](w) = [\![M]\!](\lceil w \rceil)$ を満たす yDT$^R$ $M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ を構築する．

はじめに $(s^c, \sigma^{(n)}, s^r), (s_1^c, \sigma_1, s_1^r), \ldots, (s_n^c, \sigma_n, s_n^r) \in S \times \Sigma^{(>0)} \times S$ について，次の述語 vst を考える．vst は **有効状態遷移** (*valid state transition*) の頭文字である．

$$\begin{aligned}\mathrm{vst}((s^c, \sigma^{(n)}, s^r), (s_1^c, \sigma_1, s_1^r), \ldots, (s_n^c, \sigma_n, s_n^r)) = \\ \exists s_1, \ldots, s_n \in S.\ \exists p, p_1, \ldots, p_n \in P. \\ (\forall j \in [n].\ \delta_c(s_j^c, \sigma_j) = (s_j, p_j)) \wedge \\ (\forall j \in [n-1].\ \delta_r(s_j^r, p_j, \sigma_j) = s_{j+1}^c) \wedge \\ \delta_c(s^c, \sigma) = (s_1^c, p) \wedge \delta_r(s_n^r, p_n, \sigma_n) = s^r\end{aligned}$$

vst は $(s^c, \sigma^{(n)}, s^r), (s_1^c, \sigma_1, s_1^r), \ldots, (s_n^c, \sigma_n, s_n^r) \in S \times \Sigma \times S$ が与えられたとき，$T$ において次のような状態遷移が存在するとき真になる述語である．

$$s^c \overset{\langle\sigma}{\Longrightarrow} s_1^c \overset{\langle\sigma_1}{\Longrightarrow} s_1 \Rightarrow^* s_1^r \overset{\sigma_1\rangle}{\Longrightarrow} \cdots \Rightarrow^* s_n^c \overset{\langle\sigma_n}{\Longrightarrow} s_n \Rightarrow^* s_n^r \overset{\sigma_n\rangle}{\Longrightarrow} s^r \overset{\sigma\rangle}{\Longrightarrow} s'$$

vst を用いて $M$ の先読みオートマトンの状態と遷移関数を作る．

**先読みオートマトンの状態集合**　先読みオートマトンの状態集合 $\Pi$ を次のようにして構成する．

自然数 $i \in \mathbb{N}$ によって添字付けられた集合 $\Pi_i$ を考える．$i = 0$ のとき，$\Pi_0$ をすべての $\pi = (s^c, e, s^r) \in S \times \Sigma^{(0)} \times S$ について，ある $p \in P$ が存在して $\delta_c(s^c, e) = (s^r, p)$ であるような $s^c$, $e$, $s^r$ の組を要素に持つ集合 $\Pi_0 = \{(s^c, e, s^r) \in S \times \Sigma^{(0)} \times S \mid \delta_c(s^c, e) = (s^r, p)\}$ とする．$\Pi_0$ の各要素は，$T$ において以下のような状態遷移が存在することを表している．

$$s^c \overset{\langle e}{\Longrightarrow} s^r \overset{e\rangle}{\Longrightarrow} s'$$

次に $\Pi_i$ を以下のように定義する．$\Pi_i$ は，$\Pi_{i-1}$ の要素から作ることができる vst を満たす組を $\Pi_{i-1}$ に加えた集合である．

$$\begin{aligned}&\Pi_i = \Pi_0 \cup \\ &\quad \{(s^c, \sigma^{(n)}, s^r) \in S \times \Sigma \times S \mid \exists \pi_1, \ldots, \pi_n \in \Pi_{i-1}.\ \mathrm{vst}((s^c, \sigma, s^r), \pi_1, \ldots, \pi_n)\} \quad (5.1)\end{aligned}$$

ここで $\Pi_i$ について次のような上昇列

$$\Pi_0 \subseteq \Pi_1 \subseteq \cdots \subseteq \Pi_{m-1} = \Pi_m \subseteq S \times \Sigma \times S$$

が考えられるが，$S$ と $\Sigma$ の要素数が有限であることから，$\Pi_m = \Pi_{m-1}$ を満たすような $m \in \mathbb{N}$ が存在する．そのような $m$ を用いて $\Pi = \Pi_m$ とする．これを先読みオートマトンの状態集合として使う．

**先読みオートマトンの遷移関数**　次に先読みオートマトンの遷移関数を定める．すべての $e \in \Sigma^{(0)}$ について $\theta(e) = \{(s^c, e, s^r) \in \Pi\}$ とする．また，すべての $\sigma \in \Sigma^{(>0)}$，$\pi_1, \ldots, \pi_n \in \Pi$ について，

$$\theta(\sigma^{(n)}, \pi_1, \ldots, \pi_n) = \{(s^c, \sigma, s^r) \in \Pi \mid \mathrm{vst}((s^c, \sigma, s^r), \pi_1, \ldots, \pi_n)\}$$

とする．

**変換規則**　$Q = S \times S \times \Gamma$ とする．すべての $\pi = (s^c, \sigma^{(n)}, s^r) \in \Pi$，$\gamma \in \Gamma$ について，$(s^c, \sigma, s^r) \in \theta(\sigma, \pi_1, \ldots, \pi_n)$ を満たす $\pi_1, \ldots, \pi_n \in \Pi$ が存在するときに限り，

$$(s^c, s^r, \gamma)(\sigma(x_1, \ldots, x_n)) \to \mathcal{W}((s^c, \sigma, s^r), \pi_1, \ldots, \pi_n, \gamma) \qquad \langle \pi_1, \ldots, \pi_n \rangle$$

とする．ここで $\mathcal{W}$ は $\Pi^{n+1} \times \Gamma$ から $\mathrm{Rhs}_{Q,\Delta}(X_n)$ への写像であり，次のように定義する．

$$\mathcal{W}(\pi, \pi_1, \ldots, \pi_n, \gamma) = \gamma \alpha_n^r(\beta_n^c \uplus \eta_n) \cdots \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma \in \mathrm{Rhs}_{Q,\Delta}(X_n)$$

ただし $\pi_1 = (s_1^c, \sigma_1, s_1^r), \ldots, \pi_n = (s_n^c, \sigma_n, s_n^r)$ とし，ある $s_1, \ldots, s_n \in S$ が存在し，各 $i \in [n]$ について $\delta_c(s_i^c, \sigma_i) = (s_i, p)$ であり，$\eta_i$ は $\eta_i = [\gamma := (s_i^c, s_i^r, \gamma)(x_i)]_{\gamma \in \Gamma}$ と定義される割当てとする．また，$\alpha_i^r = \rho_r(s_i^r, p_i, \sigma)$ であり，$\beta_i^c = [\gamma_p := \rho_c(s_i^c, \sigma)(\gamma)]_{\gamma \in \Gamma}$，$\alpha_\varepsilon^\Gamma = [\gamma := \varepsilon]_{\gamma \in \Gamma}$ とする．

**初期状態列の集合**　最後に初期列の集合を以下のようにする．

$$\begin{aligned} \mathit{Init} = \bigcup_{\sigma \in \Sigma} \{(F(s'')\alpha^r(s_0^r, p, \sigma)(\beta^c(\sigma) \uplus \eta(\sigma, s_0^r))\alpha_\varepsilon^\Gamma, (s_0, \sigma, s_0^r)) \mid (s_0, \sigma, s_0^r) \in \Pi. \\ \exists s', s'' \in S.\ \exists p \in P.\ \delta_c(s_0, \sigma) = (s', p) \wedge \delta_r(s_0^r, p, \sigma) = s'' \wedge s'' \in \mathrm{Dom}(F)\} \end{aligned}$$

ここで $\alpha^r(s_0^r, p, \sigma) = \rho_r(s_0^r, p, \sigma)$，$\beta^c(\sigma) = [\gamma_p := \rho_c(s_0, \sigma)(\gamma)]_{\gamma \in \Gamma}$，$\eta(\sigma, s_0^r) = [\gamma := (s_0, s_0^r, \gamma)(x_1)]_{\gamma \in \Gamma}$，$\alpha_\varepsilon^\Gamma = [\gamma := \varepsilon]_{\gamma \in \Gamma}$ とする．

□

これにより任意の SRTST と等価な変換を行う $\mathrm{yDT}^R$ を構成することができる．

ここで示した構成法により作られる $\mathrm{yDT}^R$ の例を見る．

**例 8.** 例 6 の SRTST $T$ と等価な変換を行う $\mathrm{yDT}^R$ $M$ を補題 7 で示した方法により構築する．$M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ を $\mathrm{yDT}^R$ とする．$Q = S \times \Sigma \times S$ とする．先読みオートマトンの状態集合は式 (5.1) から $\Pi_i$ を計算すると，

$$\begin{aligned} \Pi_0 &= \{(s_?, \mathtt{a}, s_?), (s_?, \mathtt{b}, s_?), (s_\mathtt{a}, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?), (s_\mathtt{b}, \mathtt{b}, s_?)\} \\ \Pi_1 &= \Pi_0 \cup \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\} \\ \Pi_2 &= \Pi_1 \cup \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\} = \Pi_1 \end{aligned}$$

となるため $\Pi = \Pi_2$ とする．遷移関数 $\theta$ は次のように定義される．

$$
\begin{aligned}
\theta(\mathtt{a}) &= \{(s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{a}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?)\},\\
\theta(\mathtt{b}) &= \{(s_?, \mathtt{b}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{b}, s_?)\},\\
\theta(\mathtt{f}, (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{a}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a})) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{a}, s_?), (s_\mathtt{b}, \mathtt{f}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{b}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{f}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{a}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{a}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{b}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{b}, s_?)) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a})) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b})) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\},\\
\theta(\mathtt{f}, (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\}.
\end{aligned}
$$

変換規則の集合 $R$ は以下の規則を含む集合である．すべての $s \in S$ について，次のように与えられる．

$$
\begin{aligned}
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{aa} && \langle (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{a}, s_?)\rangle,\\
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{ab} && \langle (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?)\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{ab} && \langle (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?)\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{bb} && \langle (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{b}, s_?)\rangle,\\
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_?, s_\mathtt{a}, \gamma)(x_1)\mathtt{a} && \langle (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{a}, s_?)\rangle,\\
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_?, s_\mathtt{a}, \gamma)(x_1)\mathtt{b} && \langle (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{b}, s_?)\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{a}(s_?, s_\mathtt{a}, \gamma)(x_1) && \langle (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{a}, s_?)\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to \mathtt{b}(s_?, s_\mathtt{a}, \gamma)(x_1) && \langle (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{b}, s_?)\rangle,\\
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_?, s_\mathtt{a}, \gamma)(x_1)(s_\mathtt{a}, s_\mathtt{a}, \gamma)(x_2) && \langle (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a})\rangle,\\
&(s, s_\mathtt{a}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_?, s_\mathtt{a}, \gamma)(x_1)(s_\mathtt{a}, s_\mathtt{b}, \gamma)(x_2) && \langle (s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b})\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_\mathtt{b}, s_\mathtt{a}, \gamma)(x_2)(s_?, s_\mathtt{b}, \gamma)(x_1) && \langle (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\rangle,\\
&(s, s_\mathtt{b}, \gamma)(\mathtt{f}(x_1, x_2)) \to (s_\mathtt{b}, s_\mathtt{b}, \gamma)(x_2)(s_?, s_\mathtt{b}, \gamma)(x_1) && \langle (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\rangle.
\end{aligned}
$$

初期列の集合は次のように定義される．

$$
\mathit{Init} = \{\langle \mathtt{a}, (s_?, \mathtt{a}, s_?)\rangle, \langle \mathtt{b}, (s_?, \mathtt{b}, s_?)\rangle, \langle (s_?, s_\mathtt{a}, \gamma)(x_1), (s_?, \mathtt{f}, s_\mathtt{a})\rangle, \langle (s_?, s_\mathtt{b}, \gamma)(x_1), (s_?, \mathtt{f}, s_\mathtt{b})\rangle\}
$$

ここで構成した $M$ に対して入力 $\mathtt{f(f(b,a),f(a,b))}$ を与えたときの計算を考える．始めに $(\Sigma, \Pi, \theta)$ において $\mathtt{f(f(b,a),f(a,b))}$ を受理する状態を求めると次のようになる．

$$\begin{aligned}
\hat{\theta}(\mathtt{a}) &= \{(s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{a}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?)\} \\
\hat{\theta}(\mathtt{b}) &= \{(s_?, \mathtt{b}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{b}, s_?)\} \\
\hat{\theta}(\mathtt{f(a,b)}) &= \{(s_?, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\} \\
\hat{\theta}(\mathtt{f(b,a)}) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{a}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\} \\
\hat{\theta}(\mathtt{f(f(b,a),f(a,b))}) &= \{(s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{a}, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{b})\}
\end{aligned}$$

この計算から $[\![M]\!](\mathtt{f(f(b,a),f(a,b))})$ を計算すると次のようになり，出力として $[\![M]\!](\mathtt{f(f(b,a),f(a,b))}) = \mathtt{abab}$ を得る．

$$\begin{aligned}
[\![M]\!](\mathtt{f(f(b,a),f(a,b))}) &= [\![\mathit{Init}((s_?, \mathtt{f}, s_\mathtt{b}))]\!](\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![(s_?, s_\mathtt{b}, \gamma)(x_1)]\!](\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![(s_?, s_\mathtt{b}, \gamma)]\!](\mathtt{f(f(b,a),f(a,b))}) \\
&= [\![\mathrm{rhs}((s_?, s_\mathtt{b}, \gamma), \mathtt{f}, \langle (s_?, \mathtt{f}, s_\mathtt{b}), (s_\mathtt{b}, \mathtt{f}, s_\mathtt{a})\rangle)]\!](\mathtt{f(b,a)}, \mathtt{f(a,b)}) \\
&= [\![(s_\mathtt{b}, s_\mathtt{a}, \gamma)(x_2)(s_\mathtt{a}, s_\mathtt{b}, \gamma)(x_1)]\!](\mathtt{f(b,a)}, \mathtt{f(a,b)}) \\
&= [\![(s_\mathtt{b}, s_\mathtt{a}, \gamma)]\!](\mathtt{f(a,b)})\,[\![(s_\mathtt{a}, s_\mathtt{b}, \gamma)]\!](\mathtt{f(b,a)}) \\
&= [\![\mathrm{rhs}((s_\mathtt{b}, s_\mathtt{a}, \gamma), \mathtt{f}, \langle (s_?, \mathtt{a}, s_?), (s_\mathtt{a}, \mathtt{b}, s_?)\rangle)]\!](\mathtt{a}, \mathtt{b}) \\
&\quad\ [\![\mathrm{rhs}((s_\mathtt{a}, s_\mathtt{b}, \gamma), \mathtt{f}, \langle (s_?, \mathtt{b}, s_?), (s_\mathtt{b}, \mathtt{a}, s_?)\rangle)]\!](\mathtt{b}, \mathtt{a}) \\
&= [\![\mathtt{ab}]\!](\mathtt{a}, \mathtt{b})\,[\![\mathtt{ab}]\!](\mathtt{b}, \mathtt{a}) \\
&= \mathtt{abab}
\end{aligned}$$

■

## 5.4 SRTST からの yDT$^R$ の構成法の正当性

本節では，5.3 節 で示した構成法が正しいことを示す．はじめに構成された yDT$^R$ の定義域が元となった SRTST の定義域が等しいことを示し，その後，SRTST の定義域において等価な変換となっていることを示す．

SRTST $T = (S, \Sigma, \Delta, P, s_0, \Gamma, F, \delta_c, \delta_r, \rho_c, \rho_r)$ とする．補題 7 の証明で示した方法により構成される yDT$^R$ を $M = (Q, \Sigma, \Delta, \mathit{Init}, R, \Pi, \theta)$ とする．

**補題 8.** すべての $\sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$ について，

$$(s^c, \Lambda, \alpha) \overset{\langle\sigma}{\Longrightarrow} (s', (p, \alpha')\Lambda, \alpha') \overset{\lfloor t_1 \rfloor}{\Longrightarrow} \cdots \overset{\lfloor t_n \rfloor}{\Longrightarrow} (s^r, (p, \alpha')\Lambda, \alpha'') \overset{\sigma\rangle}{\Longrightarrow} (s'', \Lambda, \alpha'')$$

のとき，$(s^c, \sigma, s^r) \in \Pi$ かつ $\sigma(t_1, \ldots, t_n) \in \mathcal{L}((s^c, \sigma, s^r))$ である． ■

**証明.** $\Pi$ と $\theta$ の構成法から明らかに成り立つ． □

**補題 9.** $M$ と $T$ の定義域は等しい． ■

**証明**. 補題 8 と $Init$ の構成から次が成り立つ．すべての $\sigma(t_1,\ldots,t_n)\in\mathcal{T}_\Sigma$ について，

$$
\begin{aligned}
&\langle\sigma\ \lfloor t_1\rfloor\cdots\lfloor t_n\rfloor\ \sigma\rangle\in\mathrm{Dom}(T)\\
&\iff (s_0,\varepsilon,\alpha_\varepsilon)\overset{\langle\sigma}{\Longrightarrow}_T(s',(p,\alpha_\varepsilon),\alpha)\overset{\lfloor t_1\rfloor\cdots\lfloor t_n\rfloor}{\Longrightarrow}_T(s_0^r,\alpha_\varepsilon,\alpha)\overset{\sigma\rangle}{\Longrightarrow}_T(s'',\varepsilon,\alpha')\wedge s''\in\mathrm{Dom}(F)\\
&\iff \delta_c(s_0,\sigma)=(s',p)\wedge\delta_r(s_0^r,p,\sigma)=s''\wedge s''\in\mathrm{Dom}(F)\wedge t\in\mathcal{L}((s_0,\sigma,s_0^r))\\
&\iff (\tau,(s_0,\sigma,s_0^r))\in Init\wedge t\in\mathcal{L}((s_0,\sigma,s_0^r))\\
&\iff \lceil\langle\sigma\ \lfloor t_1\rfloor\cdots\lfloor t_n\rfloor\ \sigma\rangle\rceil\in\mathrm{Dom}(M)\\
&\iff \sigma(t_1,\ldots,t_n)\in\mathrm{Dom}(M)
\end{aligned}
$$

である．□

割当て $\rho_1\in\mathcal{A}(\Gamma_1,\Gamma_1',\Delta)$，$\rho_2\in\mathcal{A}(\Gamma_2,\Gamma_2',\Delta)$ について $\Gamma_1\cap\Gamma_2=\emptyset$，$\Gamma_1'\cap\Gamma_2=\emptyset$ のとき，次の関係が成り立つ．

$$
\begin{aligned}
\rho_1\rho_2&=[\gamma_1:=\rho_1(\gamma_1)]_{\gamma_1\in\Gamma_1}\,[\gamma_2:=\rho_2(\gamma')]_{\gamma_2\in\Gamma_2}\\
&=\left[\gamma:=\begin{cases}\rho_1(\gamma) & (\gamma\in\Gamma_1\text{ の場合})\\ \gamma & (\gamma\in\Gamma_2\text{ の場合})\end{cases}\right]_{\gamma\in\Gamma_1\uplus\Gamma_2}\left[\gamma:=\begin{cases}\gamma & (\gamma\in\Gamma_1'\text{ の場合})\\ \rho_2(\gamma) & (\gamma\in\Gamma_2\text{ の場合})\end{cases}\right]_{\gamma\in\Gamma_1'\uplus\Gamma_2}\\
&=\left[\gamma:=\begin{cases}\rho_1(\gamma)\,[\gamma':=\gamma']_{\gamma'\in\Gamma_1'} & (\gamma\in\Gamma_1\text{ の場合})\\ \gamma\,[\gamma':=\rho_2(\gamma)]_{\gamma'\in\Gamma_2} & (\gamma\in\Gamma_2\text{ の場合})\end{cases}\right]_{\gamma\in\Gamma_1\uplus\Gamma_2}\\
&=\left[\gamma:=\begin{cases}\rho_1(\gamma) & (\gamma\in\Gamma_1\text{ の場合})\\ \rho_2(\gamma) & (\gamma\in\Gamma_2\text{ の場合})\end{cases}\right]_{\gamma\in\Gamma_1\uplus\Gamma_2}\\
&=\rho_1\uplus\rho_2
\end{aligned}
$$

また，ある割当て $\rho_1'\in\mathcal{A}(\Gamma_1,\Gamma_1',\Delta)$，評価 $\alpha\in\mathcal{A}(\Gamma_1',\emptyset,\Delta)$ が存在して次が成り立つ．

$$
\begin{aligned}
\rho_1\uplus\rho_2&=\rho_1\rho_2\\
&=[\gamma_1:=\gamma_1\rho_1'\alpha]_{\gamma_1\in\Gamma_1}\,\rho_2\\
&=([\gamma_1:=\rho_1'(\gamma_1)]_{\gamma_1\in\Gamma_1}\,\alpha)\rho_2\\
&=([\gamma_1:=\rho_1'(\gamma_1)]_{\gamma_1\in\Gamma_1}\uplus\rho_2)\alpha\\
&=(\rho_1'\uplus\rho_2)\alpha
\end{aligned}
$$

この結果から次のような式変形が行える．割当て $\rho^r\in\mathcal{A}(\Gamma,\Gamma\cup\Gamma_p,\Delta)$，$\rho^c\in\mathcal{A}(\Gamma,\Gamma,\Delta)$，評価 $\alpha\in\mathcal{A}(\Gamma,\emptyset,\Delta)$，$\alpha'\in\mathcal{A}(\Gamma,\emptyset,\Delta)$ について，

$$
\rho^r\,[\gamma_p:=\gamma\rho^c\alpha]_{\gamma\in\Gamma}\,\alpha'=\rho^r\,[\gamma_p:=\gamma\rho^c\alpha]_{\gamma\in\Gamma}\,[\gamma:=\psi_\gamma]_{\gamma\in\Gamma}\,[\psi_\gamma:=\alpha'(\gamma)]_{\gamma\in\Gamma}
$$

ここで，表現 $\psi_\gamma\in E(\Psi_\Gamma,\Delta)$ は $\gamma$ から唯一に定まるような表現であり，$\Gamma\cap\Psi_\Gamma=\emptyset$ とする．このとき，$[\gamma_p:=\gamma\rho_c\alpha]_{\gamma\in\Gamma}\in\mathcal{A}(\Gamma_p,\emptyset,\Delta)$，$[\gamma:=\psi_\gamma]_\gamma\in\mathcal{A}(\Gamma,\Psi_\Gamma,\Delta)$ であり，$\Gamma_p\cap\Gamma=\emptyset$ かつ $\emptyset\cap\Psi_\Gamma=\emptyset$，$\Gamma\cap\Psi_\Gamma=\emptyset$ であることから，

$$
\begin{aligned}
\rho^r\,[\gamma_p:=\gamma\rho^c\alpha]_{\gamma\in\Gamma}\,\alpha'&=\rho^r([\gamma_p:=\gamma\rho^c\alpha]_{\gamma\in\Gamma}\uplus[\gamma:=\psi_\gamma]_{\gamma\in\Gamma})\,[\psi_\gamma:=\alpha'(\gamma)]_{\gamma\in\Gamma}\\
&=\rho^r(([\gamma_p:=\rho^c(\gamma)]_{\gamma\in\Gamma}\uplus[\gamma:=\psi_\gamma]_{\gamma\in\Gamma})\alpha)\,[\psi_\gamma:=\alpha'(\gamma)]_{\gamma\in\Gamma}
\end{aligned}
$$

となる．また，$\rho^r, [\gamma_p := \rho^c(\gamma)]_{\gamma\in\Gamma} \uplus [\gamma := \psi_\gamma]_{\gamma\in\Gamma} \in \mathcal{A}(\Gamma \cup \Gamma_p, \Gamma \uplus \Psi_\gamma, \Delta)$ であることから

$$\rho^r\,[\gamma_p := \gamma\rho^c\alpha]_{\gamma\in\Gamma}\,\alpha' = \rho^r([\gamma_p := \rho^c(\gamma)]_{\gamma\in\Gamma} \uplus [\gamma := \psi_\gamma]_{\gamma\in\Gamma})\alpha\,[\psi_\gamma := \alpha'(\gamma)]_{\gamma\in\Gamma}$$

としてよい．これは，SRTST の変数の更新においてある変数を一度新しい変数で置換しておき，後で新しい変数を本来の割当てで更新しても意味が変わらない場合があるということである．また，$\psi \in \mathcal{A}(\Psi, \emptyset, \Delta)$，$\alpha \in \mathcal{A}(\Gamma, \Gamma', \Delta)$ について，$\Gamma \cap \Psi = \emptyset$ かつ $\Gamma' \cap \Psi = \emptyset$ のとき，

$$\alpha\rho = \rho\alpha$$

が成り立つ．

ここで示した割当てに関する変形を用いて，SRTST の遷移における評価を考える．

**補題 10.** $\sigma_1\ \boldsymbol{t_1}, \ldots, \sigma_n\ \boldsymbol{t_n} \in \mathcal{T}_\Sigma$ とする．各 $i \in [n]$ について，

$$\begin{aligned}(s_i^c, \Lambda, \alpha_i) &\xrightarrow{\langle\sigma_i} (s_i', (p_i, \rho(s_i^c, \sigma_i)\alpha)\Lambda', \alpha_\varepsilon)\\ &\xrightarrow{\lfloor \boldsymbol{t_{i1}} \rfloor \cdots \lfloor \boldsymbol{t_{i\,\mathrm{rank}(\sigma_i)}} \rfloor} (s_i^r, (s_i', (p_i, \rho(s_i^c, \sigma_i)\alpha)\Lambda, \alpha_i')\\ &\xrightarrow{\sigma_i\rangle} (s_i'', \Lambda, \alpha_i'')\end{aligned}$$

のとき，$\delta((s^c, \Lambda, \alpha), \langle\sigma\ \lfloor\sigma_1\ \boldsymbol{t_1}\rfloor \cdots \lfloor\sigma_n\ \boldsymbol{t_n}\rfloor) = (s^r, \Lambda', \alpha')$ とすると，

$$\alpha' = \alpha_n^r(\beta_n^c \uplus \eta_n) \cdots \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma \left( \biguplus_{i\in[n]} [(s_i^c, s_i^r, \gamma)(x_i) := \alpha_i'(\gamma)]_{\gamma\in\Gamma} \right)$$

である．ただし，各 $i \in [n]$ について $\alpha_i^r = \rho_r(s_i^r, p_i, \sigma_i)$，$\beta_i^c = [\gamma_p := \rho_c(s_i^c, \sigma_i)]_{\gamma\in\Gamma}$，$\eta_i = [\gamma := (s_i^c, s_i^r, \gamma)(x_i)]_{\gamma\in\Gamma}$ とする． ■

**証明.** $t_1 = \sigma_1\ \boldsymbol{t_1}, \ldots, t_n = \sigma_n\ \boldsymbol{t_n}$ とする．$w_1 = \lfloor \boldsymbol{t_1}1\rfloor \cdots \lfloor \boldsymbol{t_1}n\rfloor$ としたとき，遷移 $\delta((s^c, \Lambda, \alpha), \langle\sigma\ \langle\sigma_1\ w_1\ \sigma_1\rangle)$ は次のようになる．

$$\begin{aligned}(s^c, \Lambda, \alpha) &\xrightarrow{\langle\sigma} (s_1^c, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_\varepsilon)\\ &\xrightarrow{\langle\sigma_1} (s_1', (p_1, \rho_c(s_1^c, \sigma)\alpha_\varepsilon)(p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_\varepsilon)\\ &\xrightarrow{w_1} (s_1^r, (p_1, \rho_c(s_1^c, \sigma)\alpha_\varepsilon)(p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_1)\\ &\xrightarrow{\sigma_1\rangle} (s_2^c, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \rho_r(s_1^r, p_1, \sigma_1)\,[\gamma_p := \rho_c(s_1^c, \sigma_1)\alpha_\varepsilon]_{\gamma\in\Gamma}\,\alpha_1)\end{aligned}$$

ここで，$\rho_r(s_1^r, p_1, \sigma_1)\,[\gamma_p := \rho_c(s_1^c, \sigma_1)\alpha_\varepsilon]_{\gamma\in\Gamma}\,\alpha_1$ は，以下のような式変形が行える．

$$\begin{aligned}&\rho_r(s_1^r, p_1, \sigma_1)\,[\gamma_p := \rho_c(s_1^c, \sigma_1)\alpha_\varepsilon]_{\gamma\in\Gamma}\,\alpha_1\\ &= \rho_r(s_1^r, p_1, \sigma_1)\,[\gamma_p := \rho_c(s_1^c, \sigma_1)\alpha_\varepsilon]_{\gamma\in\Gamma}\,[\gamma := (s_1^r, s_1^r, \gamma)(x_1)]_{\gamma\in\Gamma}\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\\ &= \alpha_1^r([\gamma_p := \rho_c(s_1^c, \sigma_1)]_{\gamma\in\Gamma} \uplus [\gamma := (s_1^c, s_1^r, \gamma)(x_1)]_{\gamma\in\Gamma})\alpha_\varepsilon^\Gamma\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\\ &= \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\end{aligned}$$

ただし，$\alpha_1^r = \rho_r(s_1^r, p_1, \sigma_1)$，$\beta_1^c = [\gamma_p := \rho_c(s_1^c, \sigma_1)]_{\gamma\in\Gamma}$，$\eta_1 = [\gamma := (s_1^c, s_1^r, \gamma)(x_1)]_{\gamma\in\Gamma}$，$\alpha_\varepsilon^\Gamma = [\gamma := \varepsilon]_{\gamma\in\Gamma}$ とする．よって，遷移は次のようになる．

$$(s^c, \Lambda, \alpha) \xRightarrow{\langle\sigma\ \langle\sigma_1\ w_1\ \sigma_1\rangle} (s_2^c, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma})$$

同様にして，遷移 $\delta((s^c, \Lambda, \alpha), \langle\sigma\ \lfloor t_1\rfloor \cdots \lfloor t_n\rfloor)$ を考えると，

$$\begin{aligned}(s^c, \Lambda, \alpha) &\xRightarrow{\langle\sigma} (s_1^c, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_\varepsilon)\\ &\xRightarrow{\lfloor t_1\rfloor} (s_2^c, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_1^r(\beta_1^c \uplus \eta_1)\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\,\alpha_\varepsilon^\Gamma)\\ &\xRightarrow{\lfloor t_2\rfloor} \cdots\\ &\xRightarrow{\lfloor t_n\rfloor} (s_n^c, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_n^r(\beta_n^c \uplus \eta_n)\,[(s_n^c, s_n^r, \gamma)(x_n) := \alpha_n(\gamma)]_{\gamma\in\Gamma} \cdots\\ &\qquad\qquad \alpha_1^r(\beta_1^c \uplus \eta_1)\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\,\alpha_\varepsilon^\Gamma)\end{aligned}$$

となり，評価は次のように変形することができる．

$$\begin{aligned}&\alpha_n^r(\beta_n^c \uplus \eta_n)\,[(s_n^c, s_n^r, \gamma)(x_n) := \alpha_n(\gamma)]_{\gamma\in\Gamma} \cdots \alpha_1^r(\beta_1^c \uplus \eta_1)\,[(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma}\,\alpha_\varepsilon^\Gamma\\ &= \alpha_n^r(\beta_n^c \uplus \eta_n) \cdots \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma\\ &\qquad [(s_1^c, s_1^r, \gamma)(x_1) := \alpha_1(\gamma)]_{\gamma\in\Gamma} \cdots [(s_n^c, s_n^r, \gamma)(x_n) := \alpha_n(\gamma)]_{\gamma\in\Gamma}\\ &= \alpha_n^r(\beta_n^c \uplus \eta_n) \cdots \alpha_1^r(\beta_1^c \uplus \eta_1)\alpha_\varepsilon^\Gamma \left( \biguplus_{i\in[n]} [(s_i^c, s_i^r, \gamma)(x_i) := \alpha_i(\gamma)]_{\gamma\in\Gamma} \right)\end{aligned}$$

□

**補題 11.** すべての $t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma$ について，

$$(s^c, \Lambda, \alpha) \xRightarrow{\langle\sigma} (s', (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_\varepsilon) \xRightarrow{\lfloor t_1\rfloor\cdots\lfloor t_n\rfloor} (s^r, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha') \xRightarrow{\sigma\rangle} (s'', \Lambda, \alpha'')$$

のとき，すべての $\gamma \in \Gamma$ について $[\![(s^c, s^r, \gamma)]\!]_M(t) = \alpha'(\gamma)$ である． ■

**証明.** 木の構造に関する帰納法により証明する．$t_1 = \sigma_1\ \boldsymbol{t_1}, \ldots, t_n = \sigma_n\ \boldsymbol{t_n} \in \mathcal{T}_\Sigma$ について，各 $i \in [n]$ で

$$\begin{aligned}(s_i^c, \Lambda_i, \alpha_i) &\xRightarrow{\langle\sigma_i} (s_i', (p_i, \rho_c(s_i^c, \sigma_i)\alpha)\Lambda, \alpha_\varepsilon)\\ &\xRightarrow{\lfloor \boldsymbol{t_{i1}}\rfloor\cdots\lfloor \boldsymbol{t_{in}}\rfloor} (s_i^r, (p_i, \rho_c(s_i^c, \sigma_i)\alpha_i)\Lambda_i, \alpha_i')\\ &\xRightarrow{\sigma_i\rangle} (s_i'', \Lambda_i, \alpha_i'')\end{aligned}$$

のとき，すべての $\gamma \in \Gamma$ について $[\![(s_i^c, s_i^r, \gamma)]\!]_M(t) = \alpha_i'(\gamma)$ が成り立つと仮定する．$t = \sigma(t_1, \ldots, t_n)$ とする．ここで，遷移 $\delta((s^c, \Lambda, \alpha), \lfloor t\rfloor)$ が次のようなとき，

$$(s^c, \Lambda, \alpha) \xRightarrow{\langle\sigma} (s', (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha_\varepsilon) \xRightarrow{\lfloor t_1\rfloor\cdots\lfloor t_n\rfloor} (s^r, (p, \rho_c(s^c, \sigma)\alpha)\Lambda, \alpha') \xRightarrow{\sigma\rangle} (s'', \Lambda, \alpha'')$$

補題 10 より，

$$\alpha' = \alpha^r_n(\beta^c_n \uplus \eta_n) \cdots \alpha^r_1(\beta^c_1 \uplus \eta_1)\alpha^\Gamma_\varepsilon \left( \biguplus_{i\in[n]} [(s^c_i, s^r_i, \gamma)(x_i) := \alpha'_i(\gamma)]_{\gamma\in\Gamma} \right)$$

である．帰納法の仮定から

$$\alpha' = \alpha^r_n(\beta^c_n \uplus \eta_n) \cdots \alpha^r_1(\beta^c_1 \uplus \eta_1)\alpha^\Gamma_\varepsilon \left( \biguplus_{i\in[n]} [(s^c_i, s^r_i, \gamma)(x_i) := [\![(s^c_i, s^r_i, \gamma)]\!]_M (t_i)]_{\gamma\in\Gamma} \right)$$

となる．また，補題 8 より $t_1 \in \mathcal{L}((s^c_1, \sigma_1, s^r_1)), \ldots, t_n \in \mathcal{L}((s^c_n, \sigma_n, s^r_n))$ であることから，すべての $\gamma \in \Gamma$ について，

$$\alpha'(\gamma) = \mathrm{rhs}((s^c, s^r, \gamma), \sigma, \pi_1, \ldots, \pi_n) \left( \biguplus_{i\in[n]} [(s^c_i, s^r_i, \gamma)(x_i) := [\![(s^c_i, s^r_i, \gamma)]\!]_M (t_i)]_{\gamma\in\Gamma} \right)$$

となる．ただし，$\pi_1 = (s^c_1, \sigma_1, s^r_1), \ldots, \pi_n = (s^c_n, \sigma_n, s^r_n)$ とする．補題 4 より，

$$\begin{aligned} \alpha'(\gamma) &= [\![\mathrm{rhs}((s^c, s^r, \gamma), \sigma, \pi_1, \ldots, \pi_n)]\!]_M (t_1, \ldots, t_n) \\ &= [\![(s^c, s^r, \gamma)]\!]_M (\sigma(t_1, \ldots, t_n)) = [\![(s^c, s^r, \gamma)]\!]_M (t) \end{aligned}$$

である． □

**補題 12.** すべての $w \in \mathrm{Dom}(T)$ について，$[\![T]\!](w) = [\![M]\!](\lceil w \rceil)$ である． ■

**証明.** $t = \lceil w \rceil$，$t = \sigma(t_1, \ldots, t_n)$ とする．

$$(s_0, \varepsilon, \alpha_\varepsilon) \overset{\langle\sigma}{\Longrightarrow} (s', (p, \alpha_\varepsilon)\Lambda, \alpha_\varepsilon) \overset{\lfloor t_1 \rfloor \cdots \lfloor t_n \rfloor}{\Longrightarrow} (s^r_0, (p, \alpha_\varepsilon)\Lambda, \alpha') \overset{\sigma\rangle}{\Longrightarrow} (s'', \varepsilon, \alpha'')$$

$w \in \mathrm{Dom}(T)$ であることから $s'' \in \mathrm{Dom}(F)$ であり，補題 9 から $t \in \mathrm{Dom}(M)$ である．また，$[\![T]\!](w)$ は次のように変形することができる．

$$\begin{aligned} &[\![T]\!](w) \\ &= F(s'')\alpha'' \\ &= F(s'')\rho_r(s^r_0, p, \sigma)\,[\gamma_p := \gamma\rho_c(s_0, \sigma)\alpha_\varepsilon]_{\gamma\in\Gamma}\,\alpha' \\ &= F(s'')\rho_r(s^r_0, p, \sigma)\,[\gamma_p := \gamma\rho_c(s_0, \sigma)\alpha_\varepsilon]_{\gamma\in\Gamma}\,[\gamma := (s_0, s^r_0, \gamma)(x_1)]_{\gamma\in\Gamma}\,[(s_0, s^r_0, \gamma) := \alpha'(\gamma)]_{\gamma\in\Gamma} \\ &= F(s'')\rho_r(s^r_0, p, \sigma)([\gamma_p := \rho_c(s_0, \sigma)(\gamma)]_{\gamma\in\Gamma} \uplus \eta(s^r_0, \sigma))\alpha^\Gamma_\varepsilon\,[(s_0, s^r_0, \gamma) := \alpha'(\gamma)]_{\gamma\in\Gamma} \\ &= F(s'')\alpha^r(s^r_0, p, \sigma)(\beta^c(\sigma) \uplus \eta(s^r_0, \sigma))\alpha^\Gamma_\varepsilon\,[(s_0, s^r_0, \gamma)(x_1) := \alpha'(\gamma)]_{\gamma\in\Gamma} \end{aligned}$$

ここで $\alpha^r$，$\beta^c$，$\eta$，$\alpha^\Gamma_\varepsilon$ はそれぞれ $\alpha^r(s^r_0, p, \sigma) = \rho_r(s^r_0, p, \sigma)$，$\beta^c(\sigma) = [\gamma_p := \rho_c(s_0, \sigma)(\gamma)]_{\gamma\in\Gamma}$，$\eta(s^r_0, \sigma) = [\gamma := (s_0, s^r_0, \gamma)(x_1)]_{\gamma\in\Gamma}$，$\alpha^\Gamma_\varepsilon = [\gamma := \varepsilon]_{\gamma\in\Gamma}$ とする．よって，

$$[\![T]\!](w) = F(s'')\alpha^r(s^r_0, p, \sigma)(\beta(\sigma) \uplus \eta(s^r_0, \sigma))\alpha^\Gamma_\varepsilon\,[(s_0, s^r_0, \gamma)(x_1) := \alpha'(\gamma)]_{\gamma\in\Gamma}$$

である．更に補題 11 により，

$$[\![T]\!](w) = F(s'')\alpha^r(s_0^r, p, \sigma)(\beta(\sigma) \uplus \eta(s_0^r, \sigma))\alpha_\varepsilon^\Gamma\,[(s_0, s_0^r, \gamma)(x_1) := [\![(s_0, s_0^r, \gamma)]\!]_M(t)]_{\gamma\in\Gamma}$$

となる．

一方，$[\![M]\!](t)$ は定義と補題 4 から

$$\begin{aligned}[\![M]\!](t) &= [\![Init(\pi)]\!]_M(t)\\ &= [\![F(s'')\alpha^r(s_0^r, p, \sigma)(\beta(\sigma) \uplus \eta(s_0^r, \sigma))\alpha_\varepsilon^\Gamma]\!]_M(t)\\ &= F(s'')\alpha^r(s_0^r, p, \sigma)(\beta(\sigma) \uplus \eta(s_0^r, \sigma))\alpha_\varepsilon^\Gamma\,[(s_0, s_0^r, \gamma)(x_1) := [\![(s_0, s_0^r, \gamma)]\!]_M(t)]_{\gamma\in\Gamma}\end{aligned}$$

である．

以上より，$[\![T]\!](\lfloor t\rfloor) = [\![M]\!](t)$ が成り立つ． □

## 5.5 表現力

5.1 節では任意の yDT$^R$ から等価な SRTST を構成する方法を示し，5.3 節では任意 SRTST から等価な yDT$^R$ を構成する方法を示した．補題 11 と補題 7 をまとめて，次の定理を得る．

**定理 1.** SRTST と yDT$^R$ は同等の表現力を持つ． ■

補題 7 から任意の SRTST と等価な yDT$^R$ を実際に構築することができ，yDT$^R$ の等価性判定問題は決定可能 [22] であることが示されていることから次の定理が成り立つ．

**定理 2.** SRTST の等価性判定は決定可能である． ■

また，補題 7 と補題 11 で示した構成法から次の定理が言える．

**定理 3.** 任意の SRTST $T$ に対し，等価な上昇型 SRTST $T'$ を構築できる． ■

**証明.** 補題 7 から $T$ と等価な変換を定義する yDT$^R$ を $M$ が作れる．また，補題 11 から $M$ と等価な SRTST $T'$ を作れるが，これは明らかに上昇型である． □

この定理から次の系が導かれる．

**系 1.** SRTST と 上昇型 SRTST は同等の表現力を持つ． ■

# 6　関連研究

本章では，はじめに Alur と D'Antoni らによって提案された**単一使用制約付きストリーム木変換器**の単一使用制約を SRTST に適用した**単一使用制約付き木から文字列への決定性ストリーム変換器** (*streaming tree-to-string transducer with single-use-restriction*，$\mathrm{SRTST}_{\mathrm{sur}}$) の定義を示す．その後，木から文字列への変換器と文字列上の変換器における関連研究を紹介し，そこで示した変換器や SRTST ，$\mathrm{yDT}^R$ の表現力の関係について述べる．

## 6.1　単一使用制約付き木から文字列への決定性ストリーム変換器

4 章 で定義した SRTST では，変数の更新に用いる割当てにおいて変数の出現に関して制約はなかったが，ストリーム木変換器を提案した文献 [2] では，単一使用制約と呼ばれる制約が変数の更新 (割当て) に対してかけられている．ここでは，SRTST における単一使用制約について述べる．単一使用制約付きストリーム木変換器 ($\mathrm{STT}_{\mathrm{sur}}$) では，木の挿入を行うために表現に型が定められており，本論文で定義した $E(\Gamma, \Delta)$ とは異なる形となっている．単一使用制約について示す前に，変数の集合上の**衝突関係** (*conflict relation*) を考える．衝突関係 $\eta$ は変数の集合 X 上の二項関係であり，反射的かつ対称的である．衝突関係を満たす変数 $\gamma$，$\gamma'$ は同じ表現に現れることができない．

**定義 8.** $\Gamma$ を変数の集合，$\eta$ を衝突関係，$\Delta$ をアルファベットとし，表現 $e \in E(\Gamma, \Delta)$ が次を満たすとき，$e$ が $\eta$ **で一貫的である**という．

1. $e$ においてそれぞれの変数 $\gamma$ が高々一回だけ現れる．
2. $\eta(\gamma, \gamma')$ のとき，$e$ に $\gamma$ と $\gamma'$ の両方は含まない．　■

**定義 9.** $\Gamma$，$\Gamma'$ を変数の集合，$\eta$ を衝突関係，$\Delta$ をアルファベット，割当て $\rho$ が次を満たすとき**単一使用制約割当て**という．

1. 各表現 $\rho(\gamma)$ が $\eta$ で一貫的である．
2. $\eta(\gamma_1, \gamma_2)$ のとき，$\rho(\gamma_1)$ が $\gamma_1'$ を含みかつ $\rho(\gamma_2)$ が $\gamma_2'$ を含むとき，$\eta(\gamma_1', \gamma_2')$ である．■

$\Gamma$，$\Gamma'$，$\eta$，$\Delta$ からなる単一使用制約割当てすべてを含む最小の集合を $\mathcal{A}(\Gamma, \Gamma', \eta, \Delta)$ で表す．

**定義 10. 単一使用制約付きランク付き木から文字列への決定性ストリーム変換器** (*deterministic streaming ranked-tree-to-string transducer with single-use-restriction*，$\mathrm{SRTST}_{\mathrm{sur}}$) は，次のように定義される組 $(S, \Sigma, \Delta, P, s_0, \Gamma, \delta_c, \delta_r, \rho_c, \rho_r)$ である．

- $S$，$\Sigma$，$\Delta$，$P$，$s_0$，$\Gamma$，$F$，$\delta_c$，$\delta_r$ は SRTST と同様，
- $F : S \to E(\Gamma, \Delta)$ は最終出力関数であり，$F(q)$ は $\eta$ で一貫的，
- $\rho_c : Q \times \Sigma \to \mathcal{A}(\Gamma, \Gamma, \eta, \Delta)$ は呼び出し更新関数，
- $\rho_r : Q \times P \times \Sigma \to \mathcal{A}(\Gamma, \Gamma \cup \Gamma_p, \eta, \Delta)$ は戻り更新関数． ■

$\mathrm{SRTST}_{\mathrm{sur}}$ は SRTST に比べて真に表現力が弱く，**単項二階述語論理による木から文字列への変換** (*monadic second order definable tree-to-string transducer*，MSOTST) と等価な表現力を持つ [2]．MSOTST と等価な表現力を持つ計算モデルとして，**有限複製木から文字列への決定性下降型変換器** (*deterministic finite copying top-down tree-to-string transducer*，$\mathrm{yDT}_{\mathrm{fc}}$) [10] がある．

## 6.2 文字列から文字列，木から文字列への変換器

本研究で扱ったストリーム変換器は木を表すネスト文字列を入力とするが，構造を持たない通常の文字列を入力とするストリーム変換器の表現力については次のような事実が知られている．2010 年に**複製無しストリーム文字列変換器** (*copyless streaming string transducer*，$\mathrm{SST}_{\mathrm{cl}}$) [4] が Alur と Černý らによって提案された．複製無しは，更新関数を $\delta : S \times \Sigma \to \mathcal{A}(\Gamma, \Gamma, \Delta)$ としたとき，各 $s \in S$，$\sigma \in \Sigma$，$\gamma \in \Gamma$ について，文字列の集合 $\{\delta(s, \sigma)(\gamma') \mid \gamma' \in \Gamma\}$ において $\gamma$ が高々一回だけしか出現してはいけないという制約である．この複製無しという制約をストリーム木変換器上に拡張したものが単一使用制約となっている．$\mathrm{SST}_{\mathrm{cl}}$ は **単項二階述語論理による文字列変換器** (*monadic second order logic definable string transducer*, MSOST) と同等の表現力を持ち [1]，MSOST の等価性判定が可能であることから $\mathrm{SST}_{\mathrm{cl}}$ の等価性判定は決定可能である [11]．また，**複製有りストリーム変換器** (*copyful streaming string transducer*，SST) は，*HDT0L* **システム** (*deterministic 0-context lindenmayer systems with tables and with an additional homomorphism*) と等価な表現力を持つこと [14] が知られており，HDT0L システムの等価性判定が決定可能 [8, 21, 17] であることから SST の等価性判定は決定可能である．

ネスト文字列上の変換を扱う計算モデルとして，**決定性ネスト文字列から文字列への変換器** (*deterministic nested word to word transducer*，dN2W) [23] がある．dN2W は SRTST から変数を無くした計算モデルであり，スタックにはスタック記号だけを積むことができ，既に出力した文字列を後から利用することはできない．また，dN2W の等価性判定問題は文脈自由言語上の射の等価性判定へ多項式時間で帰着できることから，多項式時間で決定可能であること [23] が知られている．

yDT のサブクラスとして**木から文字列への決定性逐次下降型変換器** (*sequential yDT*，以下，逐次 yDT) がある．逐次 yDT は，すべての変換規則の右辺について各変数 $x_i$ が高々一度しか現れず，なおかつ $x_1, \ldots, x_n$ の順番で現れるような yDT のことを言う．逐次 yDT は dN2W と同等の表現力を持つこと [23] が知られている．他の yDT サブクラスとして**木から文字列への**

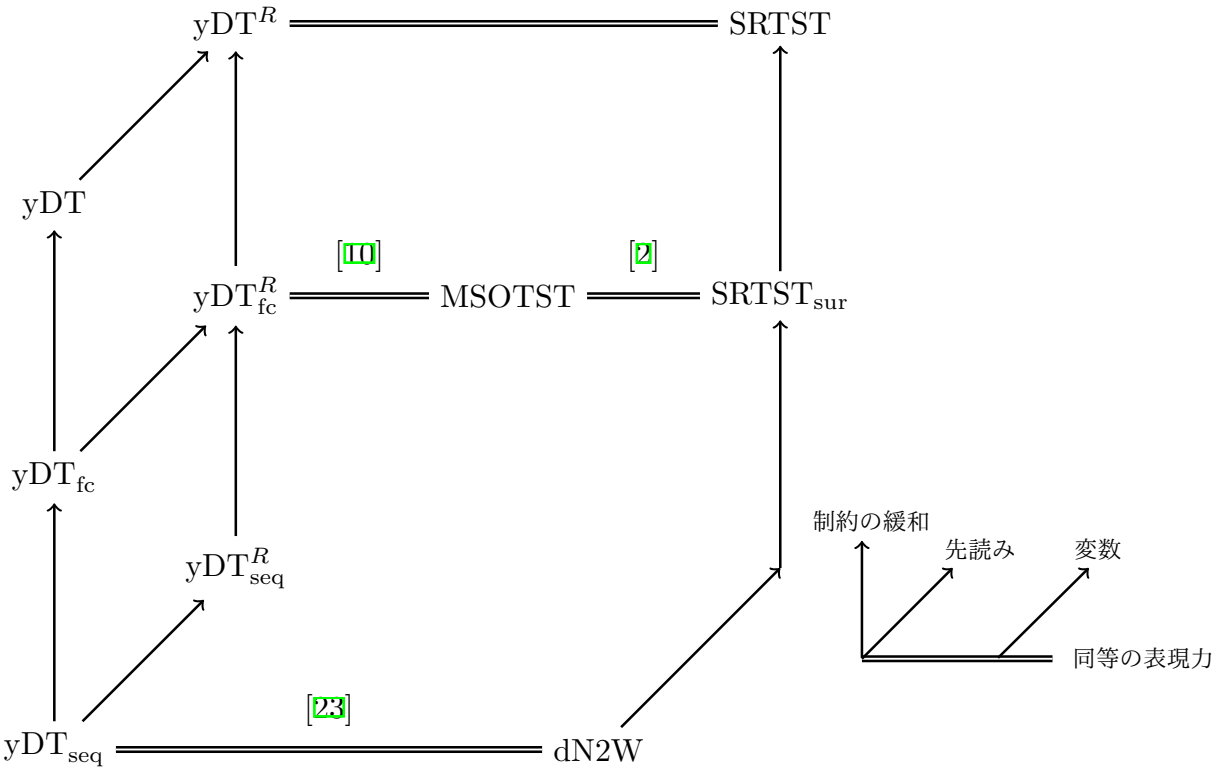


図 6.1　木から文字列への変換を定義する計算モデルの関係

横軸は同等の表現力を持つ計算モデル，左の奥向きの軸は正規先読みによる拡張，右の奥向きの軸は変数による拡張，縦軸は変数の出現に関する制約の緩和に対応する．

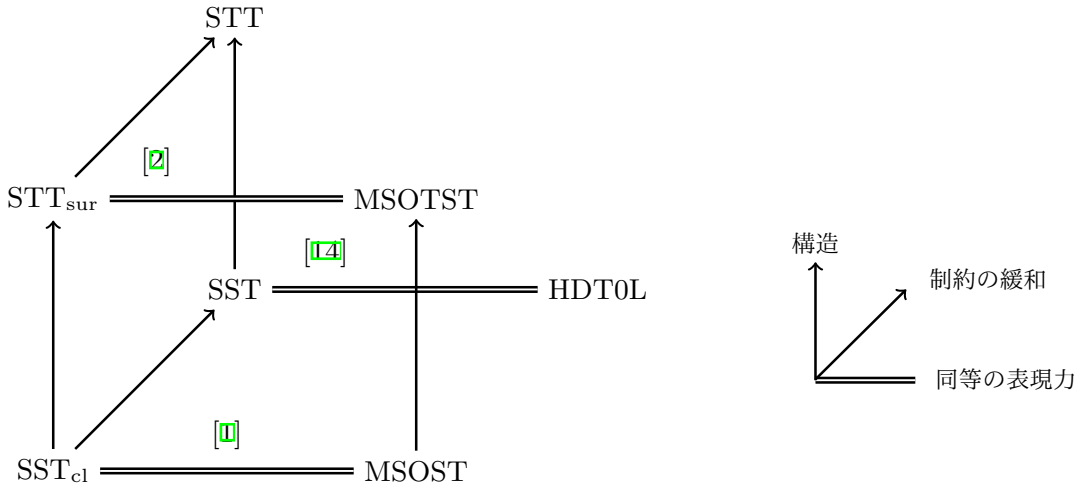


図 6.2　文字列への変換を定義する計算モデルの関係

横軸は同等の表現力を持つ計算モデル，奥向きの軸は変数の出現に関する制約の緩和，縦軸は構造を持つデータの変換への拡張 (下は入力が文字列，上は入力が木) に対応する．

**決定性線形下降型変換器** (*linear yDT*，以下，線形 yDT) がある．線形 yDT は，すべての変換規則の右辺について各変数 $x_i$ が高々一度しか現れない yDT のことをいう．線形 yDT の等価性判定問題は，多項式時間で dN2W の等価性判定問題に帰着することができる [6]．また，線形 yDT は指数時間で唯一の正規形を求めることができ，状態名の違いを除いて等価な形に規則を変形することで等価性判定を行うことができる [5]．一般の yDT については，Seidl らによって等価性判定が決定可能であることが示されている [22]．一方，yDT に累積引数を加えた**木から文字列への決定性マクロ木変換器** (*determinisitc tree-to-word macro transducer*，yDMT) の等価性判定の決定可能性は未解決問題である．

ここで挙げた計算モデルの関係は図 6.1，図 6.2 のように表すことができる．本論文で示したのは $\mathrm{yDT}^R$ と SRTST が同等の表現力を持つことである．

木の上における変換を扱うモデルとして**決定性マクロ木変換器** (*deterministic macro tree transdcuer*, MTT) がある [13]．正規先読み付き $\mathrm{MTT}^R$ が単一使用制約を満たすとき, MSOTT と同等の表現力を持つことが知られており [10], そのため $\mathrm{STT}_{\mathrm{sur}}$ と同等の表現力を持つ．MTT の等価性判定問題の決定可能性は未解決問題であるが，いくつかのサブクラスにおいて決定可能であることが知られている．詳しくはサーベイ論文である [18] を参照されたい．

構造化されたデータに対するストリーム処理に関して，**マクロ森変換器** (*macro forest transducer*) から自動で効率的なストリーム処理を導出する研究 [19] がある．マクロ森変換器は累積引数を伴う森に対する変換を扱えるため，$\mathrm{STT}_{\mathrm{sur}}$ より多くの変換を表現することができる．

# 7　おわりに

本論文では，入力をランク付きの木を表現するネスト文字列，出力を文字列に限定した決定性ストリーム木変換器 (SRTST) が正規先読み付き木から文字列への決定性降下型変換器 ($\mathrm{yDT}^R$) と同等の表現力を持つことを，等価な変換器の具体的な構成法により示した．また，それにより SRTST の等価性判定が決定可能であることと，上昇型 SRTST と SRTST が同等の表現力を持つことを示した．

本論文で用いた手法は，入力がランク付きの木であることに依存しているため，森 (ランク無し木) に対する変換も定義できる制約のない STT への応用は難しいと考えられる．しかし，STT の入力と出力をランク付きネスト文字列に制限した**決定性ストリームランク付き木変換器** (*deterministic streaming ranked-tree transducer*，SRTT) に関して，同様の手法を用いて異なる木の上の変換を扱う変換器と同等の表現力を持つことが示せる可能性がある．これについては今後の課題とする．

ここで示した $\mathrm{yDT}^R$ から SRTST への構成法により $\mathrm{yDT}^R_{\mathrm{fc}}$ から等価な SRTST を構成したとき，その SRTST は単一使用制約を必ずしも満たすとは限らない．そこで $\mathrm{yDT}^R_{\mathrm{fc}}$ から $\mathrm{SRTST}_{\mathrm{sur}}$ への変換もしくは，任意の SRTST や STT について単一使用制約を満たす形で定義可能か否かを判定する方法と，定義可能であるならばその具体的な構成法を示すことが，今後の研究の方向として挙げられる．

なお，制約のない STT の等価性判定問題の決定可能性は未解決問題として残されている [2]．

# 謝辞

本研究を行うにあたり，熱心なご指導を賜りました岩崎英哉教授，東北大学の中野圭介教授に深く感謝致します．本論文の執筆に関して助言や指摘をして頂いた東北大学の浅田和之助教に感謝致します．また，お世話になった中野研究室及び岩崎研究室の皆様に御礼申し上げます．

# 参考文献


[1] Rajeev Alur and Pavol Černý. Expressiveness of streaming string transducers. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2010)*, volume 8 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 1–12. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2010.

[2] Rajeev Alur and Loris D'Antoni. Streaming tree transducers. *J. ACM*, 64(5):31:1–31:55, August 2017.

[3] Rajeev Alur and P. Madhusudan. Visibly pushdown languages. In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 202–211. ACM, 2004.

[4] Rajeev Alur and Pavol Černý. Streaming transducers for algorithmic verification of single-pass list-processing programs. In *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '11, pages 599–610. ACM, 2011.

[5] Adrien Boiret. Normal form on linear tree-to-word transducers. In *Language and Automata Theory and Applications*, pages 439–451. Springer International Publishing, 2016.

[6] Adrien Boiret and Raphaela Palenta. Deciding equivalence of linear tree-to-word transducers in polynomial time. In *Developments in Language Theory*, pages 355–367. Springer Berlin Heidelberg, 2016.

[7] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications, 2007.

[8] K. Culik and J. Karhumäki. *A New Proof for the D0L Sequence Equivalence Problem and its Implications*, pages 63–74. Springer Berlin Heidelberg, 1986.

[9] Joost Engelfriet. Top-down tree transducers with regular look-ahead. *Mathematical systems theory*, 10(1):289–303, Dec 1976.

[10] Joost Engelfriet and Sebastian Maneth. Macro tree transducers, attribute grammars, and MSO definable tree translations. *Information and Computation*, 154(1):34 – 91, 1999.

[11] Joost Engelfriet and Sebastian Maneth. The equivalence problem for deterministic mso tree transducers is decidable. *Information Processing Letters*, 100(5):206 – 212, 2006.

[12] Joost Engelfriet, Sebastian Maneth, and Helmut Seidl. How to remove the look-ahead

of top-down tree transducers. In Arseny M. Shur and Mikhail V. Volkov, editors, *Developments in Language Theory*, pages 103–115. Springer International Publishing, 2014.

[13] Joost Engelfriet and Heiko Vogler. Macro tree transducers. *Journal of Computer and System Sciences*, 31(1):71 – 146, 1985.

[14] Emmanuel Filiot and Pierre-Alain Reynier. Copyful streaming string transducers. In *Reachability Problems*, pages 75–86. Springer International Publishing, 2017.

[15] Zoltan Fulop and H. Vogler. *Syntax-Directed Semantics: Formal Models Based on Tree Transducers*. Springer-Verlag, 1st edition, 1998.

[16] Zoltán Fülöp and Heiko Vogler. *Attributed Tree Transducers*, pages 137–171. Springer Berlin Heidelberg, 1998.

[17] Juha Honkala. A short solution for the HDT0L sequence equivalence problem. *Theoretical Computer Science*, 244(1):267 – 270, 2000.

[18] Sebastian Maneth. A survey on decidable equivalence problems for tree transducers. *International Journal of Foundations of Computer Science*, 26(08):1069–1100, 2015.

[19] Keisuke Nakano and Shin-Cheng Mu. A pushdown machine for recursive XML processing. In *Programming Languages and Systems*, pages 340–356. Springer Berlin Heidelberg, 2006.

[20] Thomas Perst and Helmut Seidl. Macro forest transducers. *Information Processing Letters*, 89(3):141 – 149, 2004.

[21] Keijo Ruohonen. *Equivalence Problems for Regular sets of Word Morphisms*, pages 393–401. Springer Berlin Heidelberg, 1986.

[22] Helmut Seidl, Sebastian Maneth, and Gregor Kemper. Equivalence of deterministic top-down tree-to-string transducers is decidable. *J. ACM*, 65(4):21:1–21:30, April 2018.

[23] Sławomir Staworko, Grégoire Laurence, Aurélien Lemay, and Joachim Niehren. Equivalence of deterministic nested word to word transducers. In *Fundamentals of Computation Theory*, pages 310–322. Springer Berlin Heidelberg, 2009.

[24] James W. Thatcher. Generalized$^2$ sequential machine maps. *Journal of Computer and System Sciences*, 4(4):339 – 367, 1970.