# 木記憶を持つ木オートマトン

6 L − 2

早川 泉[†] 中西 隆一[‡] 関 浩之[††]
† 奈良先端科学技術大学院大学 情報科学研究科 ‡ 大阪大学 基礎工学部 情報工学科

## 1 まえがき

有限オートマトンの拡張として, 木を入力とするようなオートマトンである有限状態木オートマトン (TA)[1] やプッシュダウン木オートマトン (PDTA)[2] が提案されている. TA は有限制御部のみを持ち, PDTA はさらに記憶部にプッシュダウンスタックを持つ. TA および PDTA の受理する系列言語のクラスは, それぞれ文脈自由文法およびインデックス文法の生成する言語のクラスと一致することが知られている.[1][3]

本稿ではまず, 木を入力とし, 木を記憶部に持つオートマトン (TTA) を提案する. TTA の受理する系列言語のクラスは帰納的可算な言語のクラスと一致することが示される. 次に TTA の部分クラスとして非決定性コピー TTA (NC-TTA) 等を導入する. そして, これらの部分クラスの受理する系列言語のクラスを, 有限状態変換系 (FSTS)[4] およびその部分クラスの生成する系列言語のクラスと比較する.

## 2 木記憶木オートマトン

**定義 2.1** (ボトムアップ型) 木記憶木オートマトン (frontier-to-root tree automaton with tree memory; TTA) は以下の 1~5 を満たす 5 字組 $M = (\Sigma, \Gamma, Q, F, R)$ で定義される.

1. $\Sigma = \bigcup_{i\geq 0} \Sigma_i$ は入力記号の有限集合からなる族. $\Sigma_i (i \geq 0)$ の元は $i$ 引数の入力記号を表す.
2. $\Gamma = \bigcup_{i\geq 0} \Gamma_i$ は内部記号の有限集合からなる族.
3. $Q$ は状態の有限集合.
4. $F (\subseteq Q)$ は受理状態の集合.
5. $R$ は次の形をした遷移規則の有限集合.

$$r : f((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \tau).^1$$

ただし, $f \in \Sigma_n$, $q_i\,(1 \leq i \leq n), q \in Q, \tau_i\,(1 \leq i \leq n), \tau \in T_\Gamma(X)$ である. [2] また, $\tau \in T_\Gamma(X)$ に対して, $Var(\tau)$ を $\tau$ に現れる変数全体からなる集合として, $Var(\tau) \triangleq \bigcup_{1\leq i\leq n} Var(\tau_i)$ と定義するとき, $\tau \in T_\Gamma(Var(\tau))$ でなければならない. □

TTA $M$ において, $T_\Sigma(Q \times T_\Gamma)$ 上の二項関係 $\underset{M}{\Rightarrow}$ を次のように定義する. ある遷移規則 $r : f((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \tau)$ と, ある代入 $\theta : Var(\tau) \rightarrow T_\Gamma$ が存在し, $\tau_i' = \theta(\tau_i)\,(1 \leq i \leq n)$, $\tau' = \theta(\tau)$ を満たすとき,

$$uf((q_1, \tau_1'), (q_2, \tau_2'), \cdots, (q_n, \tau_n'))v \underset{M}{\Rightarrow} u(q, \tau')v$$

とする. $\underset{M}{\Rightarrow}$ の反射推移閉包を $\overset{*}{\underset{M}{\Rightarrow}}$ と書く.

次に, $M$の受理する木言語 $T(M) \subseteq T_\Sigma$ を,

$$T(M) \triangleq \{t \in T_\Sigma | \exists q_f \in F, \exists \tau \in T_\Gamma, t \overset{*}{\underset{M}{\Rightarrow}} (q_f, \tau)\}$$

と定義する. さらに, $M$ の受理する系列言語 $yL(M)$ を,

$$yL(M) \triangleq \{yield(t) | t \in T(M)\}$$

と定義する. ただし, $t \in T_\Sigma$ に対して, $yield(t)$ を次のように定義する.

1. $a \in \Sigma_0$ に対して, $yield(a) = a$.
2. $f \in \Sigma_n\,(n \geq 1), t_i \in T_\Sigma\,(1 \leq i \leq n)$ に対して,

$$yield(f(t_1, t_2, \cdots, t_n)) = yield(t_1)yield(t_2)\cdots yield(t_n).$$

**例 2.1** $M = (\Sigma, \Gamma, Q, \{q_f\}, R)$ を次のような TTA とする. ここで, $\Sigma_0 = \{a, b, c, d\}$, $\Sigma_1 = \emptyset$, $\Sigma_2 = \{f, h\}$, $\Sigma_3 = \{g\}$, $\Sigma_n = \emptyset (n \geq 4)$, $\Gamma_0 = \{¢\}$, $\Gamma_1 = \{j\}$, $\Gamma_n = \emptyset (n \geq 2)$, $Q = \{q_a, q_b, q_c, q_d, q_l, q_r, q_f\}$ とし, $R$ に属する遷移規則を以下のように定める.

$$f((q_l, x), (q_r, x)) \rightarrow (q_f, x)$$
$$g((q_a, ¢), (q_l, x), (q_b, ¢)) \rightarrow (q_l, j(x))$$
$$g((q_c, ¢), (q_r, x), (q_d, ¢)) \rightarrow (q_r, j(x))$$
$$h((q_a, ¢), (q_b, ¢)) \rightarrow (q_l, ¢)$$
$$h((q_c, ¢), (q_d, ¢)) \rightarrow (q_r, ¢)$$
$$a \rightarrow (q_a, ¢),\ b \rightarrow (q_b, ¢),\ c \rightarrow (q_c, ¢),\ d \rightarrow (q_d, ¢)$$

このとき, $T(M) = \{f(h(a, b), h(c, d)), f(g(a, h(a, b), b), g(c, h(c, d), d)), f(g(a, g(a, h(a, b), b), b), g(c, g(c, h(c, d), d), d)), \cdots\}$, $yL(M) = \{a^n b^n c^n d^n | n \geq 1\}$ となる. □

**定理 2.1** TTA の受理する系列言語のクラスは, 帰納的可算な言語のクラスと一致する.

(略証) 任意のチューリング機械 $M$に対して, $L(M) = yL(M')$ となる TTA $M'$ を構成できることによる. □

## 3 非決定性コピー TTA

**定義 3.1** 非決定性コピー TTA (NC-TTA) とは, 以下の 1,2 を満たすような TTA $M = (\Sigma, \Gamma, Q, F, R)$ である.

1. $|F| = 1$.
2. 各遷移規則 $r : f((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \tau)$ に対して, 各 $\tau_i\,(1 \leq i \leq n)$ は変数であり, $r$ の左辺に現れる互いに異なるすべての変数のある並び換えを $x_1, x_2, \cdots, x_m\,(m \leq n)$ とするとき, ある $\delta \in \Gamma_m$ が存在して, $\tau$ は以下の形である.

$$\tau = \delta(x_1, x_2, \cdots, x_m)\ (m \leq n)$$

[1] $n = 0$ のとき, $f \rightarrow (q, \tau)$ と書く.

[2] $T_\Gamma(X)$ は, 記号集合族 $\Gamma$ と変数集合 $X$ から構成される木全体からなる集合. また, $T_\Gamma \triangleq T_\Gamma(\emptyset)$ ($\emptyset$ は空集合).

Tree Automaton with Tree Memory
Izumi Hayakawa†, Ryuichi Nakanishi‡, Hiroyuki Seki††
† Nara Institute of Science and Technology
‡ Osaka University

NC-TTA $M$ の任意の異なる遷移規則

$$r : f(\cdots) \rightarrow (q, \delta(\cdots))$$
$$r' : f'(\cdots) \rightarrow (q', \delta'(\cdots))$$

において, $q \neq q'$ または, $\delta \neq \delta'$ が成り立つとき, $M$ を**決定性コピー TTA** (DC-TTA) と呼ぶ. また, DC-TTA $M$ の任意の規則

$$r : f((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \tau)$$

の左辺において, $i \neq j\,(1 \leq i \leq n, 1 \leq j \leq n)$ ならば,「$q_i \neq q_j$ または $\tau_i \neq \tau_j$」が成り立つとき, $M$を**左線形 TTA**(LL-TTA) と呼ぶ. □

**定義 3.2** 有限状態変換系 (tree-to-string finite state translation system; FSTS) は以下の 1,2 を満たす 2 字組 $(M, G)$ で定義される.

1. $G = (N, T, P, S)$ は**文脈自由文法**. ここで, $N, T, P$ はそれぞれ, **非終端記号**, **終端記号**, $\varepsilon$ なし**生成規則**の有限集合であり, $S \in N$ は**始端記号**である.
2. $M$ は**木変換機械**と呼ばれ, 以下の (a) から (e) を満たす 5 次組 $M = (Q, \Sigma, \Delta, q_0, R)$ で定義される.
   (a) $Q$ は**状態**の有限集合.
   (b) $\Sigma = \bigcup_{i \geq 0} \Sigma_i$ は**入力記号**の有限集合からなる族. ただし, $\Sigma_n = \{p | p : A \rightarrow X_1 X_2 \cdots X_n \in P, X_i \in N \cup T (1 \leq i \leq n)\}\,(n \geq 1)$, $\Sigma_0 = T$ とする.
   (c) $\Delta$ は**出力記号**の有限集合.
   (d) $q_0 \in Q$ は**初期状態**.
   (e) $R$ は次の形をした**変換規則**の有限集合.

$$r : q[\sigma(x_1, x_2, \cdots, x_n)] \rightarrow v$$

   ただし, $q \in Q$, $\sigma \in \Sigma_n$, $v \in (\Delta \cup Q[\{x_1, x_2, \cdots, x_n\}])^*$ とする.[1]

$R$ に属する任意の異なる変換規則の左辺が互いに異なるならば, FSTS は**決定性 FSTS** (D-FSTS) と呼ばれる. また, D-FSTS における任意の変換規則の右辺において, 各 $q[x_i]\,(q \in Q$, $x_i$は左辺に現れている変数) が高々1 回しか現れないとき, D-FSTS は**右線形 FSTS** (RL-FSTS) と呼ばれる. □

FSTS $(M, G)$ において, $(\Delta \cup Q[T_\Sigma])^*$ 上の**二項関係** $\underset{M}{\Rightarrow}$ を次のように定義する. $q[\sigma(x_1, x_2, \cdots, x_n)] \rightarrow v$ を変換規則, $\alpha_1, \alpha_2 \in (\Delta \cup Q[T_\Sigma])^*$, $q \in Q$, $\sigma \in \Sigma_n$, $t_1, t_2, \cdots, t_n \in T_\Sigma$ とし, $\alpha = \alpha_1 q[\sigma(t_1, t_2, \cdots, t_n)]\alpha_2$ とする. $v$ において $x_1, x_2, \cdots, x_n$ をそれぞれ $t_1, t_2, \cdots, t_n$ で置き換えて得られる系列を $v'$ とすると,

$$\alpha \underset{M}{\Rightarrow} \alpha_1 v' \alpha_2$$

である. $\underset{M}{\Rightarrow}$ の反射推移閉包を$\overset{*}{\underset{M}{\Rightarrow}}$ と書く.

[1] $Q[\{x_1, x_2, \cdots, x_n\}] \overset{\triangle}{=} \{q[x_i] | q \in Q, 1 \leq i \leq n\}$

$\mathcal{R}(G)$ を, $G$ における構文木全体の集合と定義する. ただし, 木の各頂点のラベルは生成規則のラベルまたは終端記号であり, 根頂点のラベルは, 左辺が始端記号である生成規則のラベルとする. FSTS $(M, G)$ の生成する**系列言語** $yL(M, G)$ を,

$$yL(M, G) \overset{\triangle}{=} \{\omega \in \Delta^* | \exists t \in \mathcal{R}(G), q_0[t] \overset{*}{\underset{M}{\Rightarrow}} \omega\}$$

と定義する.

**定理 3.1** NC-TTA の受理する系列言語のクラスは, FSTS の生成する系列言語のクラスと一致する.

(略証) 任意の NC-TTA $M = (\Sigma, \Gamma, Q, \{q_f\}, R)$ に対して, $yL(M) = yL(M', G')$ を満たす FSTS $(M', G')$ を以下のように構成できることによる.(逆方向の証明は省略する.) $G' = (\{S\}, \bigcup_{i \geq 0} \Gamma_i, P', S)$, $M' = (Q, \Sigma', \Sigma_0, q_f, R')$ とする. ここで,

1. $P'$ に属する生成規則は, 各 $r : \sigma((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \delta(x_1, x_2, \cdots, x_m)) \in R\,(m \leq n)$ に対して,
$$p_{\delta,m} : S \rightarrow \underbrace{S \cdots S}_{m\,個} \delta.$$
2. $R'$ に属する遷移規則は以下の通りである.
   (a) 各 $a \rightarrow (q_a, \delta) \in R$ に対して,
$$q_a[p_{\delta,0}(x)] \rightarrow a.$$
   (b) 各 $r : \sigma((q_1, \tau_1), (q_2, \tau_2), \cdots, (q_n, \tau_n)) \rightarrow (q, \delta(x_1, x_2, \cdots, x_m)) \in R\,(m \leq n)$ に対して,
$$q[p_{\delta,m}(x_1, x_2, \cdots, x_m, x_{m+1})] \rightarrow q_1[\tau_1] q_2[\tau_2] \cdots q_n[\tau_n].$$
□

**系 3.1** DC-TTA, LL-TTA の受理する系列言語のクラスは, それぞれ, D-FSTS, RL-FSTS の生成する系列言語のクラスと一致する. □

D-FSTS の生成する系列言語は多項式時間認識可能であるので[5], 次の系が得られる.

**系 3.2** DC-TTA の受理する系列言語は多項式時間認識可能である. □

## 参考文献


[1] F.Gécseg and M.Steinby: "Tree Automata", Akadémiai Kiadó, Budapest (1984).

[2] 山崎 克典: "プッシュダウン木オートマトンと文脈自由木文法の基本的性質", 信学論 (D), **J71-D**, 9, pp.1580-1591 (1988-09).

[3] 山崎 克典: "プッシュダウン木オートマトンと Indexed 文法との関係", 信学論 (D), **J71-D**, 12, pp.2485-2497(1988-12).

[4] J.Engelfriet et al.: "Tree Transducers, L Systems, and Two-Way Machines", J. Comput. & Syst. Sci., **20**, pp.150-202 (1980).

[5] J.Engelfriet: "The Complexity of Languages Generated by Attribute Grammars", SIAM J. Comput., **15**, 1, pp.70-86 (Feb. 1986).