数理解析研究所講究録
123巻 1971年 146-169

General Indexed Grammar と Monitored Pushdown Stack Acceptor

京大 数研　西沢 輝泰

§1. 序

general indexed grammar（西沢[2]）は，直観的に言えば，各変数が'記憶'をもつような context free grammar である。生成規則は，単に変数 X が string $u_0 Y_1 u_1 \cdots Y_n u_n$（$Y_i$ は変数，u_i は terminal string）でおきかえられることを指定するだけではなく，X の記憶内容に基いて，各 Y_i の記憶内容も指定されなければならない。即ち，記憶内容の変換を表す関数 $f_1, \cdots, f_n$ を指定して，X の記憶内容が α であるとき，各 Y_i の記憶内容は $f_i(\alpha)$ であるとするのである。変数 X をどのような string でおきかえるか，またおきかえる string 内の変数の記憶内容をきめる，記憶内容の変換の指定は，ともに，X の記憶内容によって制御をうける。その制御は，記憶内容を有限種に類別しておいて，変数 X が記憶 α をもっているとき，

α が類 ξ にあれば，$X \longrightarrow u_0 Y_1 u_1 \cdots Y_n u_n$ なるおきかえが可能であり，そのとき，記憶内容の変換は，関数 $f_1, \cdots, f_n$ で指定される，という形で与される。このとき，$X \longrightarrow u_0 Y_1 u_1 \cdots Y_n u_n$ なるおきかえとそれのうける制御及び記憶内容の変換の指定は一体として，

$$(X, \xi) \longrightarrow u_0 (Y_1, f_1) u_1 \cdots (Y_n, f_n) u_n$$

なる形式で表現される。これが，general indexed grammar の生成規則である。

例として，$\{a^n b^n c^n \mid n$ は正整数$\}$ を生成する general indexed grammar を与えてみる。変数として，S, A, B, C をとり，各変数は任意の整数をその記憶内容としてとり得るものとする。記憶内容の類別，即ちこの場合整数全体の類別としては，0である，と0でないという類別を考え，前者を ξ で，後者を η であらわそう。変換 $f: \mathbb{Z} \to \mathbb{Z}$ を，$f(n) = n+1$ なる変換とし，$I: \mathbb{Z} \to \mathbb{Z}$ を恒等変換 $I(n) = n$ とする。生成規則として，

(1) $(S, \xi) \longrightarrow (S, f)$

(2) $(S, \eta) \longrightarrow (S, f)$

(3) $(S, \eta) \longrightarrow (A, I)(B, I)(C, I)$

(4) $(A, \eta) \longrightarrow a\,(A, f^{-1})$

(5) $(B, \eta) \longrightarrow b\,(B, f^{-1})$

(6) $(C,\xi) \longrightarrow c\,(C, f^{-1})$

(7) $(A,\xi) \longrightarrow a$

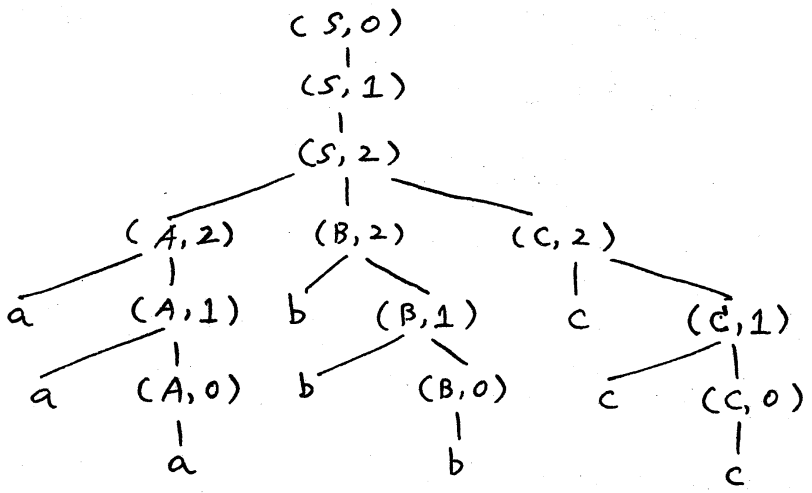
(8) $(B,\xi) \longrightarrow b$

(9) $(C,\xi) \longrightarrow c$

なる9個を与える。derivation の一例をとってみよう。sentence symbol は S, S の記憶内容は0から出発するものとしよう。記憶 α をもっている変数 X を (X,α) で表すことにすると,

$(S,0) \overset{(1)}{\Longrightarrow} (S,1) \overset{(2)}{\Longrightarrow} (S,2) \overset{(3)}{\Longrightarrow} (A,2)(B,2)(C,2)$
$\overset{(4)}{\Longrightarrow} a\,(A,1)(B,2)(C,2) \overset{(5)}{\Longrightarrow} a\,(A,1)\,b\,(B,1)(C,2)$
$\overset{(6)}{\Longrightarrow} a\,(A,1)\,b\,(B,1)\,c\,(C,1) \overset{(4)}{\Longrightarrow} aa(A,0)\,b(B,1)\,c(C,1)$
$\overset{(5)}{\Longrightarrow} aa(A,0)\,bb\,(B,0)\,c\,(C,1) \overset{(6)}{\Longrightarrow} aa(A,0)\,bb\,(B,0)\,cc(C,0)$
$\overset{(7)}{\Longrightarrow} aaa\,bb\,(B,0)\,c(C,0) \overset{(8)}{\Longrightarrow} aaa\,bbb\,cc(C,0) \overset{(9)}{\Longrightarrow} aaabbbccc$

この生成過程を tree で表現すれば,



となる。

general indexed grammar は，言語の syntax の記述を，大ワクでは context free grammar で記述しておいて，それで記述しきれない部分を，変数のもっている記憶内容で check しようとするもので，現在プログラミング言語の記述に際してとられている手法にかんがみて，十分有用であると思われる。

ところで，記憶内容は，Turing machine でいえば storage tapes にかきこまれるわけであるが，general indexed grammar でもさしあたり各変数の記憶内容は storage tapes に記憶されるものと考えよう。storage tape の構造の代数的表現としては，T. Nishizawa [1] の，general counter structure を，少し手直しして用いる。定義と例は2節で述べるが，general counter structure は，pushdown stack の構造を一般化したものである。storage tape 一本は，2本の pushdown stack で簡単に simulate されるから，general counter structure は十分に一般的な構造であり，general indexed grammar において，各種の storage tape を用いたときの共通した特性を一挙に解析することができる。

general indexed grammar なる名称は，storage tape として pushdown stack を用いると，A.V. Aho [4] の indexed grammar と生成能力が同じになることと，各変数

の記憶内容は，その変数に添えられた index とも考えられることから名づけられた。

本論では前半で 西沢[2],[3] の結果を簡単に紹介し，後半で，任意の general counter structure $\mathcal{T}$ に対し，新たな general counter structure $\mathcal{T}^*$（$\mathcal{T}$-monitored pushdown stack）が得られて，$\mathcal{T}$-indexed grammar で生成される言語は，かつそれらのみが，$\mathcal{T}^*$-counter machine（$\mathcal{T}$-monitored pda）で受理されることを示し，任意の $\mathcal{T}$ から $\mathcal{T}^*$ が得られることを利用して，general counter structures の hierarchy と，これを反映した，indexed languages の hierarchy について論ずる。

§2. General Indexed Grammars

<u>定義1.</u> general counter structure（以下 gcs と略記）とは次のような系 $\mathcal{T}=\langle \Gamma, \Pi, \mathfrak{T} \rangle$ である。

(i) Γ は空でない集合.

(ii) Π は Γ の有限分割.（$\Pi=\{\gamma_1,\cdots,\gamma_n\}$ と表すときは，γ_i は分割 Π の各ブロックであり，$\gamma_i\cap\gamma_j=\phi$（$i\neq j$），$\bigcup_{i=1}^{n}\gamma_i=\Gamma$.）

(iii) $\mathfrak{T}$ は Γ 上の変換群で，有限生成であって，かつ Γ 上推

務的。ただ Π の群演算は，$(f, g) \longmapsto g \circ f$，$g \circ f$ は $\alpha \in \Gamma$ に対し，$(g \circ f)(\alpha) = g(f(\alpha))$ なる変換とする。

例. (1) $|\Gamma| = 1$ ($|\Sigma| = 1$ でも同じ) なる gcs を trivial gcs と称し，$\boxed{0}$ で表す。

(2) $|\Gamma| < \infty$ ($|\Sigma| < \infty$ でも同じ) なる gcs を，finite gcs と称する。

(3) (counter)
$$\begin{cases} \Gamma = \mathbb{Z} \text{ (整数すべての集合)} \\ \Pi = \{ \{0\}, \mathbb{Z} - \{0\} \} \\ \Sigma = \{ \bar{n} \mid n \in \mathbb{Z} \} \quad \text{ただし，} \bar{n} \text{ は} \end{cases}$$
$\bar{n}(m) = m + n$ なる $\mathbb{Z}$ 上の変換。
この gcs を $\boxed{\frac{1}{2}}$ で表す。これは普通の counter である。

(4) (pushdown stack)
$\Gamma = \{a, b\}^{\#}$ (ただし，D を任意の集合として，$D^{\#}$ で，D で生成される自由群を表すものとする。)
$\Pi = \{ \langle e \rangle, \langle a \rangle, \langle b \rangle, \langle a^{-1} \rangle, \langle b^{-1} \rangle \}$，ただし，e は Γ の単位元で，$t \in \{e, a, b, a^{-1}, b^{-1}\}$ に対し，$\langle t \rangle = \{ x \in \Gamma \mid end(x) = t \}$; $end(x)$ は，$end(e) = e$,

$end(t_1 \cdots t_n) = t_n$, ただしここで各 t_i は a, b, a^{-1}, b^{-1} のいずれかで，各 i について $t_{i+1} \neq t_i^{-1}$ とする，により定める。

$\overline{\Gamma} = \{ \overline{x} \mid x \in \Gamma \}$, ただし，$\overline{x}$ は $\overline{x}(y) = yx$ なる Γ 上の変換（right translation）を表す。

この $gco \langle \Gamma, \Pi, \overline{\Gamma} \rangle$ を standard pushdown stack と称す。

上記の例 (3), (4) とも，分割 Π が，Γ の元唯1つの元からなるブロックをもつ。この性質は，closure property について考えるとき重要であって，gco $\tau = \langle \Gamma, \Pi, \overline{\Gamma} \rangle$ において，Π が，Γ の元唯1個からなるブロックを有するとき，τ は <u>canonical type</u> であると称することにする。

gco の合成として，最も自然なものは，2つの gco を並行して働かせる合成である。

2つの gco $\tau_i = \langle \Gamma_i, \Pi_i, \overline{\Gamma}_i \rangle$ $(i = 1, 2)$ に対し，gco $\tau = \langle \Gamma_1 \times \Gamma_2, \Pi_1 \times \Pi_2, \overline{\Gamma}_1 \times \overline{\Gamma}_2 \rangle$ を $\tau_1 \times \tau_2$ で表し，τ_1 と τ_2 の <u>product gco</u> と称す。ただしここで，$\Pi_1 \times \Pi_2$ は $\{ \xi_1 \times \xi_2 \mid \xi_1 \in \Pi_1, \xi_2 \in \Pi_2 \}$ なる $\Gamma_1 \times \Gamma_2$ の分割を表し，$(f_1, f_2) \in \overline{\Gamma}_1 \times \overline{\Gamma}_2$ は $(f_1, f_2)(\alpha_1, \alpha_2) = (f_1(\alpha_1), f_2(\alpha_2))$ なる $\Gamma_1 \times \Gamma_2$ 上の変換を表す。

定義2. gcs $\tau = \langle \Gamma, \Pi, \mathfrak{T} \rangle$ に対し，τ-indexed grammar とは，次のような系 $G = \langle V, \Sigma, P, S, \alpha \rangle$ をいう。

(i) V は nonterminal alphabet（V の元を nonterminal とも，変数とも呼ぶ），Σ は terminal alphabet，$S \in V$ は sentence symbol，α は Γ の特定の元で starting index と呼ばれる。

(ii) P は $(X, \mathfrak{x}) \longrightarrow u_0 (Y_1, f_1) u_1 \cdots (Y_n, f_n) u_n$（ただし，$X, Y_i$ は変数，$\mathfrak{x}$ は Π の元，即ち分割 Π のブロック，u_i は terminal string，f_i は $\mathfrak{T}$ の元）なる形式の rule からなる有限集合。（$n=0$ のとき，この形式は $(X, \mathfrak{x}) \longrightarrow u_0$ を表す。）

＊ τ-indexed grammar による言語の生成を定義する前に，記号の使い方の約束をおく。

τ-indexed grammar $G = \langle V, \Sigma, P, S, \alpha_0 \rangle$ に言及するときは，

(1) X, Y, Z又はこれに添字のついたものは変数を表す。

(2) a, b, c 〃 terminal 〃。

(3) u, v, w 〃 Σ^*の元（terminal string）〃。

(4) τ は特に構造が指定してなければ $\tau = \langle \Gamma, \Pi, \mathfrak{T} \rangle$ として関連事項が記述される。

(5) α, β, γ 又はこれに添字のついたものは Γ の元を表す。

(6) ξ, η 〃 Π の元 〃 。

(7) f, g 〃 重の元 〃 。

(8) φ, ψ 〃 $(\Sigma \cup (V \times \Gamma))^*$ の元〃 。

定義3. $G = \langle V, \Sigma, P, S, \alpha_0 \rangle$ を τ-indexed grammar とする。

(1) (X, α) と φ に対し，P の元 $(X, \xi) \to u_0 (Y_1, f_1) u_1 \cdots (Y_n, f_n) u_n$ で，$\alpha \in \xi$，$\varphi = u_0 (Y_1, f_1(\alpha)) u_1 \cdots (Y_n, f_n(\alpha)) u_n$ なる rule が存在するとき，$(X, \alpha) \vdash_G \varphi$ 又は単に $(X, \alpha) \vdash \varphi$ で表す。

(2) φ, ψ に対し，$\varphi = \varphi_1 (X, \alpha) \varphi_2$，$\psi = \varphi_1 \varphi_0 \varphi_2$ かつ $(X, \alpha) \vdash \varphi_0$ なる $\varphi_1, \varphi_2, (X, \alpha), \varphi_0$ が存在するとき，$\varphi \vdash_G \psi$ 又は単に $\varphi \vdash \psi$ で表す。

(3) $(\Sigma \cup (V \times \Gamma))^*$ の部分集合 A に対し，

$D(A) = \{\psi \mid \exists \varphi \in A, \varphi \vdash \psi\}$ とおく。

各 φ に対し，$D^n(\varphi)$ $(n = 0, 1, 2, \cdots)$ を，

$$\begin{cases} D^0(\varphi) = \{\varphi\} \\ D^{n+1}(\varphi) = D(D^n(\varphi)) \end{cases}$$

で定め，$D^*(\varphi) = \bigcup_{n=0}^{\infty} D^n(\varphi)$ とおく。$\psi \in D^*(\varphi)$ なるとき，φ は ψ を生成すると称し，$\varphi \overset{*}{\vdash}_G \psi$ 又は単に

$\varphi \vdash^* \psi$ で表す。

(4) G で生成される言語 $L(G)$ とは，(S, α_0) から生成される terminal string 全体である。即ち，

$$L(G) = \{u \in \Sigma^* \mid (S, \alpha_0) \vdash^* u\}.$$

(5) τ-indexed grammar で生成される言語を，τ-indexed language と称し，τ-indexed languages すべてからなる言語類を，$\mathcal{L}_\tau$ で表す。

定義4. context free grammar の特殊化として，right linear (left linear, linear) grammar を得るのと全く同様にして，right linear (left linear, linear) τ-indexed grammar を定義する。right linear τ-indexed languages すべてからなる言語類を $\mathcal{R}_\tau$ で表す。

次に [2], [3] で得られた結果（ただし＊印は [2], [3] にない新結果）を列挙する。

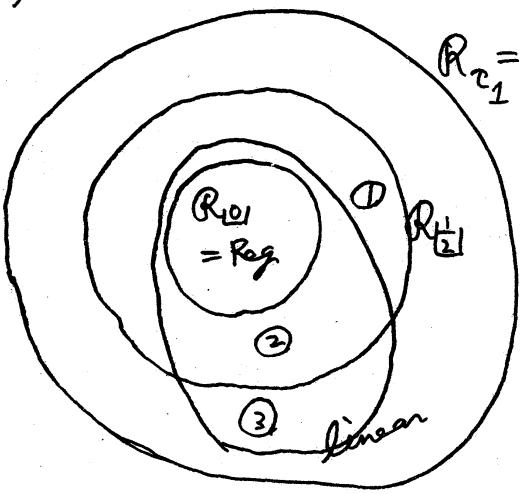
1. 以下で，τ_1 を standard pushdown stack，σ を任意の finite gcs として，

(1) $\left.\begin{aligned} \mathcal{R}_\sigma &= \{\text{regular set}\} \quad (\mathrm{Reg}) \\ \mathcal{L}_\sigma &= \mathcal{R}_{\tau_1} = \{\text{context free language}\} \quad (\mathrm{CFL}) \end{aligned}\right\}$ 自明

(2) $\mathcal{R}_{\left[\frac{1}{2}\right]\times\left[\frac{1}{2}\right]} = \mathcal{L}_{\left[\frac{1}{2}\right]\times\left[\frac{1}{2}\right]} = \{\text{recursively enumerable set}\}$ (RE)

(3) $\mathcal{L}_{\tau_1} = \{\text{Aho's indexed language}\}$ ($\mathcal{L}_{\text{index}}$)

(4)



① $\{a^n c b^n a^m c b^m \mid n, m \in \mathcal{N}\}$

② $\{a^n b^n \mid n \in \mathcal{N}\}$

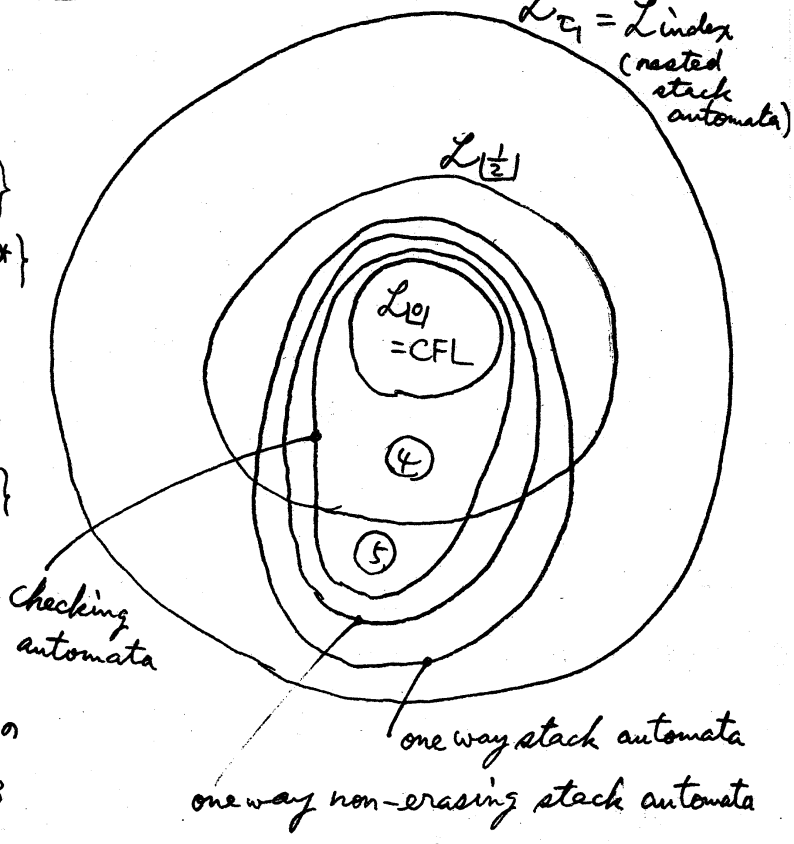
③ $\{w c w^R \mid w \in \{a,b\}^*\}$

(5)*

④ $\{a^n b^n c^n \mid n \in \mathcal{N}\}$

⑤ $\{(wc)^3 w \mid w \in \{a,b\}^*\}$

右のダイヤグラムは，言語⑤の位置（証明）と $\mathcal{L}_{\left[\frac{1}{2}\right]}$ の元 $\{a^{n^2} \mid n \in \mathcal{N}\}$ が checking automata では受理されないこと，及び S. Greibach [6] の Theorem 4.1 (p. 213) から得られる。



2. 以下で，τは一般の gcs，τ* は canonical gcs として，

(1) $\mathcal{L}_\tau \supset$ CFL, $\mathcal{R}_\tau \supset$ Reg （自明）.

(2) $\mathcal{L}_\tau$, $\mathcal{R}_{\tau^*}$ は word reversal で closed.

(3) $\mathcal{L}_\tau$ ($\mathcal{R}_\tau$) は context free languages (regular sets) を代入する substitution に関して closed.

(4) context free language (regular set) に，$\mathcal{L}_\tau$ ($\mathcal{R}_{\tau^*}$) の元を代入したものは $\mathcal{L}_\tau$ ($\mathcal{R}_{\tau^*}$) に属す。

(5) $\mathcal{L}_{\tau^*}$ は substitution で closed.

(6) $\mathcal{L}_\tau$, $\mathcal{R}_{\tau^*}$ は，union，concatenation，Kleene closure で closed. $\mathcal{R}_\tau$ は union で closed.

(7) $\mathcal{L}_\tau$, $\mathcal{R}_\tau$ は NFT mapping (A.V. Aho [4]) で closed. 従って intersection with regular set で closed.

(8) $\mathcal{L}_\tau$, $\mathcal{R}_{\tau^*}$ は full AFL.

(9) $\mathcal{L}_\tau \not\supset$ RE ($\mathcal{R}_\tau \not\supset$ RE $\wedge$ $\mathcal{R}_\tau \supset$ {Dick language}) なる，$\mathcal{L}_\tau$ ($\mathcal{R}_\tau$) は intersection で閉じていない。特に，$\mathcal{L}_{[\frac{1}{2}]}$, $\mathcal{L}_{index}$ はそうである。また $\mathcal{R}_{[\frac{1}{2}]}$ は intersection で閉じていない。

(10) $\mathcal{L}_\tau$ の元は，自然に拡張した standard function の minimal fixed point の一成分として特徴づけられる。

§3. gcs 間の simulation と，monitored pushdown stack

本節では $\mathcal{L}_{\mathrm{rt}}$ の acceptor を与える準備をする。

Notation. (1) Γ^{Γ} (Γ は空でない集合) は，$(f,g)\longmapsto g\circ f$ なる演算に関して monoid をなすが，この monoid を $\mathcal{M}(\Gamma)$ で表す。

(2) gcs $\mathrm{rt}=\langle \Gamma, \Pi, \Sigma\rangle$ に対し，$\mathcal{H}_{\mathrm{rt}}$ で次のような $(\Pi\times\Sigma)^*\longrightarrow \mathcal{M}(\Gamma)$ の homomorphism を表す。

$$\mathcal{H}_{\mathrm{rt}}((\xi,f))(\alpha)=\begin{cases} f(\alpha) & \text{if } \alpha\in\xi \\ \alpha & \text{if } \alpha\notin\xi \end{cases}$$

定義5. $\mathrm{rt}=\langle \Gamma, \Pi, \Sigma\rangle$，$\mathrm{rt}'=\langle \Gamma', \Pi', \Sigma'\rangle$ を2つの gcs とする。

(1) rt の，rt′ への representation とは，次のような系 $\mathcal{H}=(\mu,\delta,\lambda)$ である。

(i) μ は $\Gamma\longrightarrow(\mathcal{P}(\Gamma')-\{\phi\})$ の mapping.

(ii) δ は $\Pi\longrightarrow(\mathcal{P}(\Pi')-\{\phi\})$ の mapping で，$[\xi_1\neq\xi_2]\supset[\delta(\xi_1)\cap\delta(\xi_2)=\phi]$ をみたす。

(iii) λ は $(\Pi\times\Sigma)^*\longrightarrow(\Pi'\times\Sigma')^*$ の homomorphism.

(2) σ の σ' への representation $\textcircled{H} = (\mu, \delta, \lambda)$ に対し，$(\alpha, \alpha') \in \Gamma \times \Gamma'$ は，次の条件をみたすとき，$\textcircled{H}$ の下で consistent であるという。

(i) $\alpha' \in \mu(\alpha)$, (ii) $[\xi \ni \alpha \wedge \xi' \ni \alpha'] \supset [\xi' \in \delta(\xi)]$

(3) σ の，σ' への representation $\textcircled{H} = (\mu, \delta, \lambda)$ は，次の条件をみたすとき，σ, σ' による simulation と呼ばれる。

(i) 任意の $\alpha \in \Gamma$ に対し，$\alpha' \in \Gamma'$ が存在して，(α, α') は $\textcircled{H}$ の下で consistent.

(ii) 任意の $\alpha \in \Gamma$, $f \in \Xi$, $\xi \in \Pi$ に対して，(α, α') が consistent ($\alpha' \in \Gamma'$) ならば，$(\mathcal{H}_\sigma((\xi, f))(\alpha), \mathcal{H}_{\sigma'}(\lambda((\xi, f)))(\alpha'))$ は consistent.

定義 6. gcs 間の関係 $\prec$, $\sim$ を次のように定める。

(1) $\sigma \prec \sigma'$ (σ は σ' で simulate される) $\longleftrightarrow$ 適当な finite gcs δ により，σ の，$\delta \times \sigma'$ による simulation が存在する。

(2) $\sigma \sim \sigma'$ (σ は σ' に等価) $\longleftrightarrow$ $\sigma \prec \sigma' \wedge \sigma' \prec \sigma$

関係 $\prec$ について，次のことが確かめられる。

(1) 任意の gcs τ と，任意の finite gcs σ について，$\sigma \prec \tau$.

(2) $\tau_i \prec \tau_i'$ $(i=1,2)$ ならば，$\tau_1 \times \tau_2 \prec \tau_1' \times \tau_2'$.

(3) $\tau_1 \prec \tau_2$，$\tau_2 \prec \tau_3$ ならば，$\tau_1 \prec \tau_3$

(4) $\tau_1 \prec \tau_2$ ならば，$\mathcal{L}_{\tau_1} \subset \mathcal{L}_{\tau_2}$，$\mathcal{R}_{\tau_1} \subset \mathcal{R}_{\tau_2}$.
(特に，$\tau_1 \sim \tau_2$ ならば，$\mathcal{L}_{\tau_1} = \mathcal{L}_{\tau_2}$，$\mathcal{R}_{\tau_1} = \mathcal{R}_{\tau_2}$)

さて，τ- monitored pushdown stack の定義にとりかかる前に，若干の Notation を定め，$\prec$，$\sim$ について，もう少し詳しく調べる。

Notation. (1) 空でない集合 D に対する $D^\#$，及び $x \in D^\#$ に対する $end(x)$ は §2 で述べた通り。

(2) 群 $\mathfrak{G}$ に対し，$e_{\mathfrak{G}}$（又は単に e）は $\mathfrak{G}$ の単位元を表す。

(3) 群 $\mathfrak{G}$ の部分集合 M に対し，M^{-1} で $\{m^{-1} \mid m \in M\}$ を，$\overline{M}$ で $M \cup M^{-1} \cup \{e\}$ を表す。単なる空でない集合 D に対して，D^{-1}，$\overline{D}$ 等の notation を用いた場合は，D を $D^\#$ の部分集合とみなして用いたものと考える。

(4) D を空でない有限集合，$\tau = \langle \Gamma, \Pi, \mathfrak{G} \rangle$ を gcs，M を $\mathfrak{G}$ の空でない有限部分集合，α を Γ の元として，次のような gcs $\hat{\tau} = \langle \hat{\Gamma}, \hat{\Pi}, \hat{\mathfrak{G}} \rangle$ を $\otimes(D, \tau, M, \alpha)$ で表す。

(i) $\widetilde{\Gamma} = (D \times \overline{M})^{\#}$

(ii) $\widetilde{\Sigma}$ は $\widetilde{\Gamma}$ の right translations 全体のなす群。($\widetilde{\Gamma}$ と同一視する。)

(iii) φ を，$\widetilde{\Gamma} \to \widetilde{\Sigma}$ の次のような homomorphism とする。(a) $\varphi((\alpha, f)) = f$ for all $(\alpha, f) \in D \times \overline{M}$，(b) $\varphi(t_1 \cdot t_2) = \varphi(t_2) \circ \varphi(t_1)$ for all $t_1, t_2 \in \widetilde{\Gamma}$.

$t \in (D \times \overline{M}) \cup (D \times \overline{M})^{-1}$ と $\xi \in \Pi$ に対し，$J(t, \xi)$ で $\{ x \in \widetilde{\Gamma} \mid end(x) = t,\ \varphi(x)(\alpha) \in \xi \}$ を表す。これにより，$\widetilde{\Pi} = \{ J(t, \xi) \mid \xi \in \Pi,\ t \in (D \times \overline{M}) \cup (D \times \overline{M})^{-1} \} \cup \{ \{ e_{\widetilde{\Gamma}} \} \}$.

補助定理 7. $\tau = \langle \Gamma, \Pi, \Sigma \rangle$，$M, M'$ を Σ を生成する Σ の(有限)部分集合，α, α' を Γ の任意の元とする。

(1) D, D' を，2個以上の元をもつ任意の有限集合とすれば，$\otimes(D, \tau, M, \alpha) \sim \otimes(D', \tau, M', \alpha')$

(2) τ が nontrivial gcs であれば，D, D' を任意の空でない有限集合として，$\otimes(D, \tau, M, \alpha) \sim \otimes(D', \tau, M', \alpha')$

(3) $\tau = \langle \{\alpha\}, \{\{\alpha\}\}, \{I\} \rangle$ を trivial gcs (101)，D を2個以上の元をもつ有限集合，D_0 を唯1個の元からなる集合とすれば，

(a) $\otimes(D, \tau, \{I\}, \alpha) \sim$ standard pushdown stack

(b) $\otimes(D_0, \tau, \{I\}, \alpha) = \boxed{\frac{1}{2}}$

定義8. gcs $\tau = \langle \Gamma, \Pi, \Xi \rangle$ に対し，M は Ξ を生成する Ξ の有限部分集合，α は Γ の元，D は空でない有限集合とするとき，gcs $\otimes(D, \tau, M, \alpha)$ を τ-monitored pds (pushdown stack) と称す。

上の補助定理7によって，

1. τ が nontrivial gcs なら，τ-monitored pds $\widetilde{\tau}$ は，同等なものを同一視すれば唯1つである。
2. $\boxed{0}$-monitored pds は同等なものを同一視すれば唯2つある。1つは $\boxed{\frac{1}{2}}$ であり，他の1つは standard pds に同等なものである。後者を $\boxed{1}$ で表す。

補助定理9. (1) ${}^*\tau$ を nontrivial gcs，τ を一般の gcs として，$({}^*\tau \times \tau)^{\sim} \prec {}^*\tau^{\sim} \times \tau$.

(2) ${}^*\tau$ を nontrivial gcs として，${}^*\tau^{\sim} \prec {}^*\tau \times \boxed{1}$

定理10. (1) τ が nontrivial gcs なら，$\boxed{1} \prec \widetilde{\tau}$.

(2) τ_1, τ_2 が nontrivial gcs で，$\tau_1 \prec \tau_2$ なら，$\widetilde{\tau_1} \prec \widetilde{\tau_2}$. 特に，$\tau_1 \sim \tau_2$ なら，$\widetilde{\tau_1} \sim \widetilde{\tau_2}$.

(3) σ が nontrivial finite gcs なら, $\widetilde{\sigma} \sim \boxed{1}$.

定理 10(2) により, 次に定める $\boxed{n}$, $\boxed{n\frac{1}{2}}$ はそれぞれ同等なものを除いて唯1つである。

定義 11.
$$\begin{cases} \boxed{2} = \boxed{1}^{\sim}, & \boxed{1\frac{1}{2}} = \boxed{\frac{1}{2}}^{\sim} \\ \boxed{n+1} = \boxed{n}^{\sim}, & \boxed{(n+1)\frac{1}{2}} = \boxed{n\frac{1}{2}}^{\sim}. \end{cases}$$

明らかに, $k \leqq k'$ ならば $\boxed{k} < \boxed{k'}$ である。

§4. $\mathfrak{R}_\tau$, $\mathfrak{L}_\tau$ の acceptor.

まず, $\mathfrak{R}_\tau$ の acceptor は次のように自然に定義される。

定義 12. gcs $\tau = \langle \Gamma, \Pi, \Xi \rangle$ に対する τ-counter machine (τ-CM) とは, 次のような系 $\mathfrak{P} = \langle \Sigma, K, \delta, q_0, \alpha_0, F \rangle$ である。

(i) Σ は input alphabet, K は state alphabet, q_0 は initial state ($q_0 \in K$), F は set of final states ($\subset K$), α_0 は starting counter content ($\alpha_0 \in \Gamma$).

(ii) δ は，$K\times\Pi\times(\{\varepsilon\}\cup\Sigma)$ の各元に，$K\times$更 の空でない有限部分集合を対応させる，$K\times\Pi\times(\{\varepsilon\}\cup\Sigma)\longrightarrow\mathcal{P}(K\times$更$)$ の mapping．ただし，ε は monoid Σ^* の単位元を表す。

定義 13. $\mathbb{P}=\langle\Sigma, K, \delta, q_0, \alpha_0, F\rangle$ をτ-CM とする．

(1) $K\times\Gamma\times\Sigma^*$ 上の2項関係 $\vdash_{\mathbb{P}}$ を次のように定める。

$(q,\alpha,u)\vdash_{\mathbb{P}}(q',\alpha',u') \longleftrightarrow [\exists a\in\{\varepsilon\}\cup\Sigma,\ \exists f\in$更 ; $\delta(q,\xi,a)\ni(q',f)$ for $\xi\ni\alpha\ \wedge\ f(\alpha)=\alpha'\ \wedge\ u=au']$

(2) $\vdash_{\mathbb{P}}$ の transitive, reflexive closure を $\vdash^*_{\mathbb{P}}$ とする．

(3) $T(\mathbb{P})=\{u\in\Sigma^*\mid (q_0,\alpha_0,u)\vdash^*_{\mathbb{P}}(q,\alpha,\varepsilon)$ for some $(q,\alpha)\in F\times\Gamma\}$ を，$\mathbb{P}$ で受理(accept)される set (又は language) という．何らかのτ-CM で受理される language をτ-CM language という。

定理 14. τ-CM language $\longleftrightarrow$ right linear τ-indexed language.

定義 15. τ-monitored pushdown stack acceptor (τ-monitored pda) とは τ-CM をいう。τ-CM lan-

guage を τ-monitored pda language とよぶ。

定理 16. τ-indexed language ⟷ τ-monitored pda language (i.e. $\mathcal{L}_\tau = \mathbb{R}_{\tilde{\tau}}$).
(この定理は context free language ⟷ pda language の一般化である。)

定理16により, nested stack automaton (A.V. Aho [4], [5]) と 1-monitored pda とが同等であることがわかる。これ自体, 興味ある結果である。

1-monitored pda は, 具体的には, 次のような2本の pushdown stack (pds) D と S を storage tapes としてもつ, on-line nondeterministic Turing machine である。即ち, D に対しどのような symbol を pushdown 又は popup するかは, state と D, S の top symbols によって定まるが, S に対しどのような symbol を pushdown 又は pop-up するかは, そのとき D に対しどのような symbol が push-down 又は pop-up されるかに完全に従属して定まる。しかもこの従属関係は, D に symbol A を pushdown することが S に symbol B を pushdown することであるならば, D から A を pop-up することが即 S から B を pop-up することにな

ような，制限の強い従属関係である。このようなpds Dとその上との関係を，その上はDのslaveである，と表現することにする。すると，$|n|$-monitored pdaとは，具体的には，次のような$n+1$本のpdsをstorage tapesとしてもつ，on-line nondeterministic Turing machineである。即ち，2本目のpdsが1本目のpdsのslave，……，$i+1$本目のpdsがi本目のpdsのslave，……，$n+1$本目のpdsがn本目のpdsのslaveである。又，$|n\frac{1}{2}|$-monitored pdaは，$|n|$-monitored pdaにさらにもう1本storage tapeとしてcounterを持ったもので，このcounterは$n+1$本目のpdsのslaveになっているようなTuring machineである。

§4. Hierarchy

前節定理16と§2の結果から，直ちに次の定理を得る。

定理17. $\mathcal{R}_{|0|} = \text{Reg} \subsetneq \mathcal{R}_{|\frac{1}{2}|} \subsetneq \mathcal{R}_{|1|} = \mathcal{L}_{|0|} = \text{CFL} \subsetneq \mathcal{R}_{|1\frac{1}{2}|} = \mathcal{L}_{|\frac{1}{2}|} \subsetneq \mathcal{R}_{|2|} = \mathcal{L}_{|1|} = \mathcal{L}_{\text{index}} \subset$

$$\mathcal{R}_{\lfloor 2\frac{1}{2}\rfloor} \subset \cdots\cdots \subset \mathcal{R}_{\lfloor n\rfloor} \subset \mathcal{R}_{\lfloor n\frac{1}{2}\rfloor} \subset \mathcal{R}_{\lfloor n+1\rfloor} \subset \cdots\cdots .$$
$$\mathcal{R}_{\lfloor n\rfloor} = \mathcal{L}_{\lfloor n-1\rfloor} \qquad \mathcal{R}_{\lfloor n\frac{1}{2}\rfloor} = \mathcal{L}_{\lfloor (n-1)\frac{1}{2}\rfloor} \qquad \mathcal{R}_{\lfloor n+1\rfloor} = \mathcal{L}_{\lfloor n\rfloor}$$

従って，

$$\lfloor 0\rfloor \lneq \lfloor \tfrac{1}{2}\rfloor \lneq \lfloor 1\rfloor \lneq \lfloor 1\tfrac{1}{2}\rfloor \lneq \lfloor 2\rfloor \lneq \lfloor 2\tfrac{1}{2}\rfloor \lneq \cdots \lneq \lfloor n\rfloor \lneq \lfloor n\tfrac{1}{2}\rfloor \lneq \lfloor n+1\rfloor \lneq \cdots\cdots .$$

Notation. alphabet Σ の各元 a に対し，Σ の元でない新しい symbol $\#_a$ を，$a \neq a'$ ならば $\#_a \neq \#_{a'}$ となるように対応させる。$u, v \in \Sigma^*$ に対し $u^{(v)}$ なる string を，$u^{(\varepsilon)} = \varepsilon$，$u^{(a)} = u\#_a u\#_a u\#_a u$ $(a \in \Sigma)$，$u^{(xa)} = u^{(x)} \cdot u^{(a)}$ $(x \in \Sigma^*, a \in \Sigma)$ により定める。

補助定理18. $L_1 = \{ u\#(ucu^{(v)}\#)^2 u \mid v = a^{|u|}b^{|u|}, u \in \{a,b\}^* \}$ は $\lfloor 1\rfloor$-indexed language ではない。(ただし，$\#$，c は $\#_a, \#_b$ とは異なる2つの symbols とする。)

補助定理19. $L_2 = \{ u\#(ucucuc\,u^{(u)}\#)^2 u \mid u \in \{a,b\}^* \}$ は $\lfloor 1\frac{1}{2}\rfloor$-indexed language ではない。(ただし，$\#$，c については上記と同じ。)

補助定理 18, 19 の証明では，前者で $\{ucu^{|u|}b^{|u|} \mid u \in$

$\{a,b\}^*\}$ が context free language ではないことを，後者では，$\{u\#u\#u\#u \mid u\in\{a,b\}^*\}$ が $\boxed{\frac{1}{2}}$ - indexed language ではないことを用いる。

$L_1 \in \mathcal{L}_{\boxed{1\frac{1}{2}}}$，$L_2 \in \mathcal{L}_{\boxed{2}}$ は実際に grammar を構成することによって示すことができ，次の定理が成立する。

定理 20. $\mathcal{L}_{\boxed{2}} \supsetneqq \mathcal{L}_{\boxed{1\frac{1}{2}}} \supsetneqq \mathcal{L}_{\boxed{1}}$

系 21. $\boxed{0} \not\leq \boxed{\frac{1}{2}} \not\leq \boxed{1} \not\leq \boxed{1\frac{1}{2}} \not\leq \boxed{2} \not\leq \boxed{2\frac{1}{2}} \not\leq \boxed{3}$.

定理21により，我々は $\mathcal{L}_{\text{index}}$ より真に大きく，RE より真に小さい full AFL $\mathcal{L}_{\boxed{1\frac{1}{2}}}$ を得ることができた。

次の問題が未解決である。

1. $\forall n \geqq 2$ に対し，$\mathcal{L}_{\boxed{n}} \subsetneqq \mathcal{L}_{\boxed{n\frac{1}{2}}} \subsetneqq \mathcal{L}_{\boxed{n+1}}$?
2. 1が成立しない場合，$\exists n_0$, $\mathcal{L}_{\boxed{k}} \subset \mathcal{L}_{\boxed{n_0}}$ for $\forall k$?
3. $\bigcup_{n=0}^{\infty} \mathcal{L}_{\boxed{n}} \subsetneqq \mathrm{RE}$?
4. $\mathcal{L}_{\boxed{n}}$ $(n \geqq 1\frac{1}{2})$ に関する member-ship problem.

($\mathcal{L}_{\boxed{n}}$ に関しては，member-ship problem, emptiness problem, $\varepsilon \in L$? の3問題群はいずれか1つが可解であれば残りも可解である。)

参考文献

[1] T. Nishizawa, Sequential Machines with General counters, Mem. Fac. Sci., Kyushu Univ., Ser A, vol. 24, no. 1, 94-99 (1970)

[2] 西沢輝泰, General Indexed Grammars, 日本数学会応用数学分科会予稿集 (1970, 4月)

[3] ————, General Indexed Grammar に関する若干の補足, 日本数学会応用数学分科会予稿集 (1970, 10月)

[4] A. V. Aho, Indexed Grammars — An Extension of Context-Free Grammars, J. ACM, vol. 15, no. 4, 647-671 (1968)

[5] ————, Nested Stack Automata, J. ACM, vol. 16, no. 3, 383-406 (1969)

[6] S. Greibach, Checking Automata and One-Way Stack Languages, J. CSS, vol. 3, 196-217 (1969)