数理解析研究所講究録
第 322 巻 1978 年 20-55

Simple Loop Program による time complexity の階層

京大 数理研 笠井 琢美
日本IBM 足立 暁生

§1 序

従来の loop program による階層は、実際的計算可能性という見地からみて広すぎる。loop のネストの深さが 2 の loop program によって実現される関数のクラス $\mathcal{L}_2$ が初等関数のクラスと一致してしまう。ここでは loop program を制限することによって time complexity のもっと細かい階層を研究する。即ち simple loop program という loop program の制限されたクラスを導入する。これによって2種類の階層 $\mathcal{I}_0 \subset \mathcal{I}_1 \subset \cdots$ と $\mathcal{E}^0_{xp} \subset \mathcal{E}^1_{xp} \subset \cdots$ を syntactical に charactalize する。ここで $\mathcal{I}_k$ は $O(n^k)$ 時間で認識できる言語のクラス、$\mathcal{E}^k_{xp}$ は n の k 重の指数関数時間で認識できる言語のクラスである。ここで n は入力の長さである。したがって $\bigcup_{k=0}^{\infty} \mathcal{I}_k = \mathcal{E}^0_{xp}$ は多項式時間認識可能な言語のクラス、$\bigcup_{k=0}^{\infty} \mathcal{E}^k_{xp}$ は初等的問題のク

ラスである。

一般にプログラムの time complexity は計算可能でない。しかし、ここで述べるように syntax を制限すれば計算量のよい上界が、そのプログラムを syntactical に解析することによって得ることができる。

§2. loop program on SM

<u>Def.</u> X, X′, X″を互いに素な可算集合とする。X の元を <u>control variable</u>, X′の元を <u>variable</u>, X″の元を <u>array name</u> と呼ぶ。<u>loop program</u> とは次で定義される statement のことをいう。ここで i, j は variable, x は control variable, u は variable か又は control variable, A は array name, c は $N = \{0, 1, \cdots\}$ の元を表わす。

< atomic statement > ::= $u \leftarrow u+1$ | $u \leftarrow u \dot{-} 1$

| $u \leftarrow c$ | $A[u] \leftarrow v$ | $i \leftarrow A[u]$

< loop statement > ::= loop x do < statement > end

< condition > ::= $u = c$ | $u \neq c$

< statement > ::= < atomic statement > | < loop statement >

| if < condition > then < statement > else < statement > end

| < statement > ; < statement >

<u>Def.</u> <u>ステップ機（SMと略す）のプログラム</u>とは列

$$\alpha_1 ; \alpha_2 ; \cdots\cdots ; \alpha_k$$

のことをいう。ここで α_i は次のいづれか

(i) do <atomic statement> then goto l

(ii) if <condition> then goto l else goto l'

(iii) halt,

但し $1 \leq l, l' \leq k$.

<u>Def.</u> $\tilde{X}$ で変数および添字付変数の集合を表わす。すなわち，$\tilde{X} = X \cup X \cup \{A[i] : A \in X'', i \in N\}$。関数 $d : \tilde{X} \to N$ を <u>memory map</u> と呼ぶ。memory map 全体からなる集合を D と書く。P を loop program（SMのプログラム）とする。すると P は D から D への関数（部分関数）を実現する。P の実現する関数を同じ記号 $P : D \to D$ で表わす。P の time complexity とは関数 $time_P : D \to N$ で，$time_P(d)$ は初期 memory d のもとで P が実行する atomic statement の個数を表わす。

<u>Remark.</u> Loop program は SM のプログラムの制限されたものとみることができる。ここでの loop program は本来のもの[3, 4, 9] とは多少異なる。従来の loop program

では $u \leftarrow u \dot{-} 1$ の形のステートメント，及び if-then-else を含んでいないし，配列も許していない。
$u \leftarrow u \dot{-} 1$ と if-then-else を許した理由は，これらをなしで表現するためには，loop が 1 個余分に必要とするからであり，配列を許した理由は，配列を含むプログラムを配列を含まないプログラムで"多項式時間"では simulate できないからである[7]。

Def. P を loop program とする。このとき X 上の関係 $>_P$ を次のように定める。$x >_P y$ であるとは，P が loop x do Q end の形の statement を含み，Q が $y \leftarrow y+1$ を含むときをいう。P が simple であるとは

$$x_1 >_P x_2 >_P \cdots >_P x_k, \quad k > 1, \quad x_1 = x_k$$

なる列を含まないときをいう。したがって P が simple なら $>_P$ の reflexive transitive closure は X 上の partial order である。

Def. P を loop program，s を P に現われる atomic statement の occurrence とする。このとき s の深さとは s を含む loop statement の個数をいう。$s_1, \cdots s_k$ を P の深さ ≥ 1 の atomic statement の occurrence とする。

t_i を A_i の深さとする。このとき $t_1, t_2 \cdots, t_k$ を P の <u>loop complexity</u> と呼び, $l_c(P)$ で示す。

<u>例1.1.</u>　P:　loop x do
　　　loop y do　$z \leftarrow z+1$　end;
　　　loop x do
　　　　loop x do　$y \leftarrow y+1$　end end end

とする。すると $x >_P y$, $y >_P z$. よって P は simple であり, $l_c(P) = 2 \times 3 = 6$ である。

以下で loop complexity によって時間量の上界が計算できることを示す。また多項式時間計算可能な言語のクラスと loop complexity との関係について調べる。§5 ではプログラム P の時間量のもっと正確な解析をする方法について述べる。

§3　基本的性質

<u>Def.</u> プログラム P が <u>reduced</u> であるとは, P が $x \leftarrow x+1$, x は control variable, の形の statement と loop statement だけから構成されるときをいう。P を program とするとき, P の <u>reduced form</u> $\bar{P}$ を次のように定める。

zを（どんなプログラムにも）現われない control variable とする。 Pが $x \leftarrow x+1$ の形以外の atomic statement なら $\bar{P}$ は $z \leftarrow z+1$, Pが if q then P_1 else P_2 なら $\bar{P}$ は $\bar{P}_1; \bar{P}_2$, Pが loop x do P_1 end なら $\bar{P}$ は loop x do $\bar{P}_1$ end。

Def. 各 $k \in N$ に対し、$S_k : D \to D$ を次のように定める。 各 $x \in \tilde{X}$ に対し

$$S_k(d)(x) = \begin{cases} d(x) & \text{if } d(x) \geq k \\ n & \text{otherwise} \end{cases}$$

d, d' を D の元とする。 任意の $x \in X$ に対し、$d(x) \leq d'(x)$ となるとき $d \leq d'$ と書く。

Lemma 3.1 Qを reduced な loop program とする。

(i) 任意の $d \in D$ に対し、$d \leq Q(d)$

(ii) $d \leq d'$ なら $Q(d) \leq Q(d')$

(iii) $d \leq d'$ なら $\mathrm{time}_Q(d) \leq \mathrm{time}_Q(d')$

Proof. 明らか。

Lemma 3.2 Pを loop program、$\bar{P}$をPの reduced form とすると (i) $l_c(P) = l_c(\bar{P})$, (ii) $k \in N$ が存

在し任意の $d \in D$ に対し

$$\mathrm{time}_p(d) \leq \mathrm{time}_{\bar{p}}(S_k(d))$$

<u>Proof</u> (i)は $\bar{P}$ の構成より明らか $\bar{P}$ の構成による帰納法で，(ii)と次の (3.1) を示す。

(3.1) $k \in N$ が存在し任意の $d \in D$ に対し

$$P(d) \leq \bar{P}(S_k(d))$$

が成立する。 P が atomic statement なら (ii) 及び (3.1) は明らかに成立。（P が $x \leftarrow c$, $c \in N$, のとき，$k = c$ とおくと (3.1) が成立する）

P が if q then P_1 else P_2 end のとき，帰納法の仮定より，k_1, k_2 が存在し

$$\mathrm{time}_{p_i}(d) \leq \mathrm{time}_{\bar{p}_i}(S_{k_i}(d))$$

$$P_i(d) \leq \bar{P}_i(S_{k_i}(d))$$

が成立する。 $k = \max\{k_1, k_2\}$, $d' = \bar{P}_1(S_k(d))$ とおくと Lemma 3.1 より

$$S_k(d) \leq d'$$

よって

$$\begin{aligned}\mathrm{time}_p(d) &= \max\{\mathrm{time}_{p_1}(d), \mathrm{time}_{p_2}(d)\}\\ &\leq \mathrm{time}_{p_1}(d) + \mathrm{time}_{p_2}(d)\\ &\leq \mathrm{time}_{\bar{p}_1}(S_k(d)) + \mathrm{time}_{\bar{p}_2}(S_k(d))\end{aligned}$$

$$\leq time_{\bar{p}_1}(S_k(d)) + time_{\bar{p}_2}(d')$$
$$= time_{\bar{p}_1;\bar{p}_2}(S_k(d)) = time_{\bar{p}}(S_k(d))$$

よって(ii)が成立する。　$d \in D$, $x \in X$ に対し

$$P(d)(x) = \max\{P_1(d)(x),\ P_2(d)(x)\}$$
$$\leq \max\{\bar{P}_1(S_k(d))(x),\ \bar{P}_2(S_k(d))\}$$
$$\leq \bar{P}_2(\bar{P}_1(S_k(d))(x) = \bar{P}(S_k(d))(x)$$

よって (3.1) が成立。

P が loop x do P_1 end のとき, $d_0 \in D$, $d_0(x) = t$ とし,

$$d_{i+1} = P_1(d_i),$$
$$d_0' = S_k(d_0), \quad d_{i+1}' = \bar{P}_1(d_i'), \quad 0 \leq i < t$$

とおく。帰納法の仮定より k が存在し

$$P_1(d_i) \leq \bar{P}_1(S_k(d_i))$$
$$time_{p_1}(d_i) \leq time_{\bar{p}_1}(S_k(d_i))$$

が成立する。i に関する帰納法でます

$$(3.2) \quad S_k(d_i) \leq d_i'$$

を示す。$i = 0$ のときは明らか。i で成立するとする。

$$S_k(d_{i+1}) = S_k(P_1(d_i))$$
$$\leq S_k(\bar{P}_1(S_k(d_i)))$$
$$= \bar{P}_1(S_k(d_i)) \leq \bar{P}_1(d_i') = d_{i+1}'。$$

よって (3.2) はすべての i, $0 \leq i < t$ で成立する。よって

$$time_p(d_0) = \sum_{i=0}^{k-1} time_{p_1}(d_i)$$

$$\leq \sum_{i=0}^{k-1} time_{\bar{P}}(S_k(d_i))$$

$$\leq \sum_{i=0}^{k-1} time_{\bar{P}_1}(d_i') = time_{\bar{P}}(d_0)$$

よって(ii)が成立。 また

$$P(d_0) = d_t \leq S_k(d_t) \leq d_t' = \bar{P}(d_0)$$

より(3.1)が成立。 $P = P_1 ; P_2$ のときも同様である。

<u>Def.</u> PとQをreducedなloop programとする。このとき任意の $d \in D$ に対し $P(d) \leq Q(d)$ となるとき $P \leq Q$ と書く。 Pがreducedなら

$$time_P(d) = \sum_{x \in X} (P(d)(x) - d(x))$$

である。 よって次が成立する。

<u>Lemma 3.3</u> P, Qをreducedとする。 $P \leq Q$ なら任意 $d \in D$ に対し

$$time_P(d) \leq time_Q(d)$$

<u>Lemma 3.4</u> P, Q, Q′をreducedとし, $Q \leq Q'$ とする。 PがQを含むとき, PのQをQ′で置き替えて得られるprogramをP′とする。 すると

$P \leq P'$。

<u>Def.</u> P を loop program とする。 $x \in X$ とする。
P が $x \leftarrow x+1$ を含むなら x は P の <u>right variable</u> といい、P が loop$x$ do Q end の形の statement を含むとき x は P の <u>control variable</u> という。 P の right varible 全体からなる集合を $Rv(P)$, control variable 全体からなる集合 $Cv(P)$ と書く。

<u>Lemma 3.5</u> P, Q を reduced な loop program とする。

$Rv(P) \cap Cv(Q) = \phi$ なら

$P ; Q \leq Q ; P$

<u>Proof.</u> R を reduced な loop program とする。このとき R の構成に関する帰納法で次を示すことができる。

(3.3) 任意の $x \in Cv(R)$ に対し $d(x) \leq d'(x)$ なら任意の $y \in X$ に対し

$$R(d)(y) - d(y) \leq R(d')(y) - d'(y).$$

$d_0 \in D$, $d_1 = P(d_0)$, $d_2 = Q(d_1)$, $d_1' = Q(d_0)$, $d_2' = P(d_1')$ とおく。 すると任意の $x \in Cv(Q)$ に対し $d_0(x) = d_1(x)$ であるから (3.3) より任意の $y \in X$ に対し

(3.4) $d_2(y) - d_1(y) = d_1'(y) - d_0(y)$

が成立する。 また任意の $x \in Cv(P)$ に対し $d_0(x) \leq d_1'(x)$

が成立するから，(3.3)より任意の$y \in X$に対し

(3.5) $d_1(y) - d_0(y) \le d_2'(y) - d_1'(y)$

が成立する。(3.4)と(3.5)より

$$d_2(y) \le d_2'(y) \quad \text{for all} \quad y \in X$$

となり $d_2 \le d_2'$ となる。

<u>Def.</u> loop program の <u>normal form</u> を次のように定義する。ここで x は control variable を表わす。

<type 1> ::= $x \leftarrow x+1$
| loop x do <type 1> end

<normal form> ::= <type 1> | <normal form>; <normal form>

<u>Theorem 3.6</u> P を reduced な simple loop program とする。このとき normal form の simple loop program Q で次を満たすものが存在する。

(i) $l_c(P) = l_c(Q)$ (ii) $P \le Q$

<u>Proof.</u> P が次の形の statement を含んだとする

loop x do $P_1; P_2 \ldots, P_k$ end, $k > 1$

まず

(3.7) $Cv(P_i) \cap Rv(P_j) = \phi$ for all j, $i<j\leq k$

と仮定してよいことを示す。 P_iは simple であるから、
あるiが存在し $i<j$ なる任意のj ($1\leq j\leq k$) に対し

$$Cv(P_i) \cap Rv(P_j) = \phi$$

が成立する。 Lemma 3.4 と 3.5 より

$$P_1; P_2; \cdots; P_k \leq P_i; P_1; \cdots P_{i-1}; P_{i+1}; \cdots, P_k.$$

よって左辺を右辺に置きかえたものを新たにPとすればよい。

(3.7) が成立したとする。 このとき

(3.8) loop x do $P_1; \cdots; P_k$ end

$\leq$ loop x do P_1 end; loop x do $P_2; \cdots$

$; P_k$ end。

が成立することを示す。 $d\in D$, $d(x)=t$ とする。
すると Lemma 3.5 より

loop x do $P_1; \cdots; P_k$ end

$= (P_1; \cdots; P_k)^t \leq P_1^t; (P_2; \cdots; P_k)^t$

$=$ loop x do P_1 end; loop x do $P_2; \cdots; P_k$ end.

が成立する. 上述の操作をくり返すことによって目的のQが得られる。

<u>Theorem 3.7</u> Pを simple loop program, $d\in D$,

$m = \max_{x \in X} d(x)$ とおく。 すると定数 c が存在し

$$\mathrm{time}_P(d) \le c\, m^{l_c(P)}$$

<u>proof</u> P の reduced form を $\bar{P}$ とする。 P が simple なら $\bar{P}$ も simple. Lemma 3.2 より $l_c(P) = l_c(\bar{P})$ で

$$(3.8) \quad \mathrm{time}_P(d) \le \mathrm{time}_{\bar{P}}(S_k(d)) \text{ for some } k.$$

Theorem 3.6 と Lemma 3.3 より, normal form の simple loop program Q が存在し $l_c(\bar{P}) = l_c(Q)$ で

$$(3.9) \quad \mathrm{time}_{\bar{P}}(d) \le \mathrm{time}_Q(d)$$

が成立する。 よって任意の normal form の simple loop program Q に対し

$$(3.10) \quad \mathrm{time}_Q(d) \le c\,(\max_x d(x))^{l_c(Q)} \quad \text{for some } c.$$

が示せればよい。 なぜなら (3.8), (3.9), (3.10) より

$$\begin{aligned}
\mathrm{time}_P(d) &\le \mathrm{time}_{\bar{P}}(S_k(d)) \\
&\le \mathrm{time}_Q(S_k(d)) \\
&\le c\,(\max_x d(x) + k)^{l_c(Q)} \\
&\le c'\,(\max d(x))^{l_c(P)}
\end{aligned}$$

が成立するからである。 (3.10) を Q の構成に関する帰納法で示す。 Q が type 1 なら明らかに成立する。 $Q = Q_1 ; Q_2$, $l_c(Q_i) \ge 1$ とする。 $d' = Q_1(d)$ と

おく。 すると

$$\max_x d'(x) \leq m + \mathrm{time}_{Q_1}(d)$$
$$\leq m + c_1 m^{l_1(Q_1)}$$

よって

$$\mathrm{time}_Q(d) = \mathrm{time}_{Q_1}(d) + \mathrm{time}_{Q_2}(d')$$
$$\leq c_1 m^{l_c(Q_1)} + c_2(m + c_1 m^{l_c(Q_1)})^{l_c(Q_2)}$$
$$\leq c_3\, m^{l_c(Q_1)\cdot l_c(Q_2)} \quad \text{for some } c_3$$

となり (3.10) が成立する。

§4 Polynomial time computability

多くの問題は言語の認識問題として形式化される。$[k] = \{1, 2, \cdots, k-1\}$ を alphabet, $[k]^*$ の subset を言語と呼ぶ。 言語 $L \subset [k]^*$ が SM のプログラム P で $f(n)$ 時間認識可能であるとは, P に $n a_1 a_2 \cdots a_n$, $n \in N$, $a_i \in [k]$, が入力として与えられたとき $f(n)$ 時間以内に P が停止し, $a_1 \cdots a_n \in L$ なら 1 を そうでないなら 0 を出力するときをいう。 (たとえば, 変数 X に入力の長さ n が, 配列 $A[1], \cdots, A[n]$ に $a_1, \cdots, a_n$ が set された状態で P は計算を開始し, 停止したときの変数 Y の値が 1 なら受理, 0 なら拒否とする。) L が多テープ Turing Machine (TM と略す) で $f(n)$ 時間認識可

能なら，L は SM で $O(f(n))$ 時間認識可能なら，L は TM で $O(f^2(n))$ 時間認識可能である。

<u>Notation</u>　各 $k \in N$ に対し，P_k で SM で $O(n^k)$ 時間認識可能な言語のクラスを表わす。$P = \bigcup_{k=0}^{\infty} P_k$ とおく。したがって P は多項式時間認識可能な言語のクラスを表わす。　次にプログラムのクラスを simple loop program に制限しても，時間に関しては，それほど不経済とはならないことを示す。　証明は本質的に Constable and Bordin に負う。

<u>Lemma 4.1</u>　L が SM のプログラム P で cn^k 時間認識可能であったとする。　すると L は loop complexity が k である simple loop program Q で受理できる。

<u>Proof</u>　simple loop program P_i, $1 \leq i \leq k$, を次のように定める。

$$P_0 = y \leftarrow y+1$$

$$P_{i+1} = \text{loop } x \text{ do } P_i \text{ end}$$

また R を次のように定める。

$$y \leftarrow 0;\quad z \leftarrow 0;\quad P_k;\ (\text{loop } y \text{ do } z \leftarrow z+1 \text{ end};)^{c}$$

$$d' = (y \leftarrow 0;\ z \leftarrow 0;\ P_k)(d),\quad d'' = R(d),\quad d(x) = n$$

とすると $\alpha'(y)=n^k$, $\alpha'(z)=\ulcorner c \urcorner, n^k$ となる。 Pは次の3つの型の statement の有限集合である

(4.1) i: do α then goto l

(4.2) i: if q then goto l else goto l'

(4.3) i: halt

ここで α は atomic statement, q は判定である。 また P は control variable を含まないとしてよい。 Qは次のように定められる。

R; $w \leftarrow 0$; loop z do R_0 end

ここで 各 R_i は次のように定められる。 P の i 番目の statement が (4.1) の形なら

R_i = if $w=0$ then α; $w \leftarrow l$ else $w \leftarrow w \dot{-} 1$; R_{i+1} end.

P の i 番目の statement が (4.2) の形なら

R_i = if $w=0$ then
if q then $w \leftarrow l$ else $w \leftarrow l'$ end
else $w \leftarrow w \dot{-} 1$; R_i+1

(4.3) の形なら

R_i = if $w=0$ then $w \leftarrow i$ else $w \leftarrow w \dot{-} 1$; R_{i+1} end.

このように構成されたQは明らかにPと同値である。

また $l_c(Q) = l_c(P_k) = k$。

Lemma 4.1 と Theorem 3.7 より次の定理が得られる。この定理は $\mathcal{P}_k$ の一つのプログラムによる syntactic charactarization を与える。

<u>Theorem 4.2</u>　$L \in \mathcal{P}_k$ である必要十分条件は loop complexity k の simple loop program で L が受理されることである。

<u>Corollary 4.3</u>　$L \in \mathcal{P}$ である必要十分条件は L が simple loop program で受理されること。

§5　精密な解析

loop complexity により simple loop program の時間量のおおざっぱな上界が得られる。たとえば例1.1のプログラム P を考える。$d \in D$, $n = \max_x d(x)$ とおくと P の loop complexity は 6 だから

$$\mathrm{time}_P(d) \leq cn^6$$

となる。しかし、もっと正確には

$$\mathrm{time}_P(d) \leq cn^4$$

がいえる。ここでは与えられた simple loop program の時間量の上界をもっと正確に求める方法について述べる。

<u>Def</u> Pを解析しようとするプログラムとする。X_PでPに現われる control variable の集合とする。このとき <u>Pのterm</u> を次のように定義する。ここでcはNの元 xはX_Pの元を表わす。

$$\langle term \rangle ::= c \mid x \mid (\langle term \rangle \cdot \langle term \rangle)$$
$$\mid (\langle term \rangle + \langle term \rangle) \mid \max\{\langle term \rangle, \dots, \langle term \rangle\}$$

$d \in D$, tをtermとする。このとき $d(t) \in N$ を次のように定義する。

$$d(c) = c, \qquad c \in N$$
$$d(t_1 + t_2) = d(t_1) + d(t_2)$$
$$d(t_1 \cdot t_2) = d(t_1) \cdot d(t_2)$$
$$d(\max\{t_1, \dots, t_m\}) = \max\{d(t_1), \dots, d(t_m)\}$$

$X_P = \{x_1, \dots, x_n\}$ とする。ここで Pの各substatement Qに対し、以下の性質(5.1), (5.2)を持つようなtermの列 $V(Q)$, $U(Q)$ を求める algorithm について考える。

(5.1) $V(Q)$はtermのn組$(t_1, \dots, t_n)$で任意の$d \in D$, $x_i \in X_P$に対し

$$Q(d)(x_i) \leq d(t_i)$$

が成立する。

(5.2) Qがk個の atomic statement を持つなら，Qは term のk組$(u_1, \cdots, u_k)$で，任意の$d \in D$に対し，dのもとでQを実行したとき，Qのi番目の atomic statement が実行される回数$d(u_i)$回以内である。さらに次の algorithm で構成される$V(Q)$は次の性質を持つ。

(5.3) $V(Q)$のi番目の component t_iは次の形。

$\max\{v_1, \cdots, v_m\}$

ここで各v_j, $1 \leq j \leq m$, は変数x_iを含まないか，$v_j = x_i$ があるいは$v_j = x_i + v_j'$の形で，v_j'はx_iを含まない。さらにv_jがx_ℓ $(\ell \neq i)$を含めば$x_i <_P x_\ell$である。特に$m=1$のとき，$\max\{v_1\}$のかわりにv_1と書く。

<u>Algorithm 5.1</u>

<u>Input</u>: simple loop program P

<u>Output</u>: (5.2) を満す$U(P)$

<u>方法</u>: P の各 substatement Q に対し，(5.1)〜(5.3) を満す$V(Q)$, $U(Q)$を順に構成していく。$X_P = \{x_1, \cdots, x_n\}$とする。

<u>Case 1</u> Q が atomic statement のとき

$$U(Q) = \{1\}$$

$V(Q) = (t_1, \cdots, t_n)$ は次のように定められる。

Case 1.1　Q が $x_i \leftarrow x_i + 1$ のとき

$$t_j = \begin{cases} x_j & \text{if } j \neq i \\ x_i + 1 & \text{if } j = i \end{cases}$$

Case 1.2　Q が $x_i \leftarrow c$, $c \in N$ のとき

$$t_j = \begin{cases} x_j & \text{if } j \neq i \\ c & \text{if } j = i \end{cases}$$

Case 1.3　Q がその他の atomic statement のとき

$$t_j = x_j \qquad 1 \leq j \leq n$$

Case 2　Q = if q then Q_1 else Q_2 end のとき

$$U(Q) = U(Q_1) \cup U(Q_2)$$

と定める。$V(Q_1) = (\max u_1, \max u_2, \cdots, \max u_n)$, $V(Q_2) = (\max v_1, \cdots, \max v_n)$ とする。このとき $V(Q)$ を

$$V(Q) = (\max u_1 \cup v_1, \max u_2 \cup v_2, \cdots, \max u_n \cup v_n)$$

と定める。

Case 3. $Q = Q_1 ; Q_2$ のとき、$V(Q_1) = (u_1, \cdots, u_n)$
$V(Q_2) = (v_1, \cdots, v_n)$ とする。 t を任意の term とするとき、$\tilde{t}$ で t に現われる x_i を u_i で置き替えて得られる term を表わす。 $U(Q)$ は次で定められる。

$$U(Q) = U(Q_1) \cup \{\tilde{t} : t \in U(Q_2)\}$$

$V(Q) = (t_1, \cdots, t_n)$ は次のように定義される。 $u_i = \max\{u_{i1}, \cdots, u_{ij_i}\}$, $v_i = \max\{v_{i1}, \cdots, v_{ik_i}\}$ とする。

$$t_i = \max(\{\tilde{v}_{if} : v_{if} \text{ が } x_i \text{ を含まない。} 1 \le f \le k_i\}$$
$$\cup \{u_{ig} : 1 \le g \le j_i, \text{ if } v_{if} = x_i \text{ for some } f\}$$
$$\cup \{u_{ig} + \tilde{w}_f : 1 \le g \le j_i, v_{if} = x_i + w_f \text{ の形のとき}\})$$

Case 4 $Q = \text{loop } x_\ell \text{ do } Q_1 \text{ end}$ のとき、$V(Q_1) = (u_1, u_2, \cdots, u_n)$ より $V(Q) = (t_1, \cdots, t_n)$ を求める。ここで $<_p$ の大きい順に t_i を計算する。すなわち $x_i <_p x_j$ なら t_j を t_i より先に計算する。

今、$x_i <_p x_j$ なるすべての j に対し、t_j が求まったとする。 $u_i = \max\{v_1, \cdots, v_f, x_i + w_1, x_i + w_2, \cdots x_i + w_g\}$ 又は $u_i = \max\{v_1, \cdots, v_f, x_i + w_1, \cdots, x_i + w_g, x_i\}$ とする。ただし $f \ge 0$, $g \ge 0$ で v_ℓ, w_ℓ は x_i を含まない term。

このとき t_i を

$$t_i = \max\{\max\{\tilde{v}_1, \cdots, \tilde{v}_f\} + x_\ell \cdot \max\{\tilde{w}_1, \cdots, \tilde{w}_q\},$$
$$x_i + x_\ell \cdot \max\{\tilde{w}_1, \cdots \tilde{w}_q\}\}$$

と定める。 ここで $\tilde{w}$ は term より，w に現われる各変数 x_j を t_j で置き替えて得られる term．（v_ℓ 又は w_ℓ が変数 x_j を含めば (5.3) より $x_i <_P x_j$． よって t_j はすでに求まっている。） $U(Q)$ は次で定められる。

$$U(Q) = \{x_\ell \cdot \tilde{t} : t \in U(Q_1)\}$$

<u>Algorithm 5.2</u>

<u>Input</u> : simple loop program P

<u>Output</u> : term t such that, for all $d \in D$,
$time_P(d) \leq t(d)$

<u>方法</u> : $U(P)$ を Algorithm 5.1 で計算し
$$t = \sum_{t' \in U(P)} t'$$ とする。

例 5.3　次のプログラム P を考える

```
y ← 0;  z ← 0;  w ← 0;
loop x do
   if A[y] = 0 then
      w ← 3;
      loop y do
         z ← z+1; w ← w+1 end
   else
      loop z do  w ← w+1 end
   end;
   loop w do  B[w] ← w end;
   y ← y+1
end
```

P は simple である。次に Algorithm 5.1 に従って $V(P_i)$, $U(P_i)$ を計算する。$V(P_i)$ の変数 x に対応する要素が t ならば $x \leftarrow t$ と書く。また $x \leftarrow x$ の形は省略する。

$V(P_1) = \{z \leftarrow z+1,\ w \leftarrow w+1\}$, $U(P_1) = \{1, 1\}$

$V(P_2) = \{z \leftarrow z+y,\ w \leftarrow w+y\}$, $U(P_2) = \{y, y\}$

$V(P_3) = \{z \leftarrow z+y,\ w \leftarrow 3+y\}$, $U(P_3) = \{1, y, y\}$

$V(P_4) = \{w \leftarrow w+z\}$, $U(P_4) = \{z\}$

$V(P_5) = \{w \leftarrow \max\{3+y,\ w+z\},\ z \leftarrow z+y\}$,

$U(P_5) = \{1, y, y, z\}$

$V(P_6) = \{y \leftarrow y+1\}$, $U(P_6) = \{w, 1\}$

$$\mathcal{V}(P_7) = \{w \leftarrow \max\{3+y, w+z, w+z\}, z \leftarrow z+y, y \leftarrow y+1\} \quad \mathcal{U}(P_7) = \{1, y, y, z, w, 1\}$$

$$\mathcal{V}(P_8) = \{y \leftarrow y+x, \; z \leftarrow z+x(y+x), \; w \leftarrow \max\{3+y+x, w\}+x(z+x)(y+x)\}$$

($w <_p z <_p y$ であるから、y, z, w, の順に計算する)

$$\mathcal{U}(P_8) = \{x, x(y+x), x(y+x), x(z+x(y+x)), x(\max\{3+y+x, w\}+x(z+x(y+x))), x\}$$

よって求める $\mathcal{U}(P)$ は次のようになる

$$\mathcal{U}(P) = \{x, x^2, x^2, x^3, x(3+x+x^3), x, 1, 1, 1\}$$

よって

$$time_P(d) \leq d(x^4+x^3+3x^2+5x+3)$$

<u>Theorem 5.4</u> Algorithm 5.1 は $\mathcal{U}(P)$ を正しく計算する。

<u>Proof</u> Q の構成に関する帰納法で $\mathcal{U}(Q)$, $\mathcal{V}(Q)$ が (5.1)〜(5.3) を満たすことを証明する。Case 1, Case 2 の場合は明らかである。

Case 3 の場合を考える。構成より $\mathcal{V}(Q)$ が (5.3) を満たすことは明らか。(v_i が x_j, $j \neq i$ を含み u_j が x_i を含むことはない。もしそうだとすると、$x_i <_p x_j$,

$x_j <_P x_i$ となり，P が simple であることに矛盾する。)

$v_{if} = x_i + w$ とする $1 \leq f \leq k_i$．すると

$$\tilde{v}_{if} = \max\{u_{i1}, \ldots, u_{j_i}\} + \tilde{w}$$
$$= \max\{u_{i1} + \tilde{w}, \ldots, u_{j_i} + \tilde{w}\}$$

よって

$$t_i = \max\{\tilde{v}_{i1}, \ldots, \tilde{v}_{ik_i}\} = \tilde{v}_i$$

となる。$d' = Q_1(d)$，$d'' = Q_2(d)$ とする。

帰納法の仮定より

$$d'(x_j) \leq d(u_j), \qquad 1 \leq j \leq n$$
$$d''(x_i) \leq d'(v_i) \qquad 1 \leq i \leq n$$

よって

$$d''(x_i) \leq d(\tilde{v}_i) = d(t_i)$$

となり (5.1) が成立する。(5.2),(5.3) が成立することは明らか。

Case 4 を考える。t_i が変数 x_ℓ を含むのは $q \neq 0$ のとき．このとき u_i は $x_i + u_i$ という term を含む。

したがって Q_1 は $x_i \leftarrow x_i + 1$ という atomic statement を含み，$x_i <_P x_\ell$ となる。よって (5,3) が成立。

任意の $d \in D$ と $m \in N$ に対し．$d_m \in D$ を

$$d_m(x_i) = \begin{cases} d(x_i) & \text{if } i \neq \ell \\ m & \text{if } i = \ell \end{cases}$$

と定義する。また

$$d'_m = Q_1^m(d)$$

とする。(5.1) を示すには、任意の i, $i \neq l$, と m に対し

$$(5.4) \qquad d'_m(x_i) \le d_m(t_i)$$

が成立することを言えばよい。($i = l$ なら、Q は statement $x_l \leftarrow x_l + 1$ を含まないから、$Q(d)(x_l) \le d(x_l) \le d(t_l)$ が成立する。)

$x_i <_p x_j$ なる任意の j に対し、(5.4) が成立したと仮定する。m に関する induction で (5.4) を示す。$m = 0$ なら $d'_m(x_i) = d_m(x_i) \le d_m(t_i)$。よって (5.4) は成立する。m で成立すると仮定する。任意の term t に対し、$d \le d'$ なら $d(t) \le d'(t)$。

よって特に

$$d_m(t) \le d_{m+1}(t)$$

$d'_{m+1} = Q_1(d'_m)$ より

$$\begin{aligned} d'_{m+1}(x_i) &\le d'_m(\max\{\max\{v_1, \cdots, v_f\}, \\ &\qquad x_i + \max\{w_1, \cdots, w_g\}, x_i\}) \\ &\le \max\{\max\{d'_m(v_1), \cdots, d'_m(v_f)\}, \\ &\qquad d'_m(x_i) + \max\{d'_m(w_1), \cdots, d'_m(w_g)\}, \\ &\qquad d'_m(x_i)\} \end{aligned}$$

今 $x_i <_P x_j$ なる任意の j に対し

$$dm(x_j) \leq dm(t_j)$$

が成立する。よって

$$dm(v_\ell) = dm(\tilde{v}_\ell),\ dm(w_\ell) \leq dm(\tilde{w}_\ell).$$

帰納法の仮定より

$$\begin{aligned} dm(x_i) &\leq dm(t_i) \\ &\leq dm(\max\{\max\{\tilde{v}_1, \cdots, \tilde{v}_p\} + x_\ell \max\{\tilde{w}_1, \cdots, \tilde{w}_q\}, \\ &\qquad x_i + x_\ell \max\{\tilde{w}_1, \cdots, \tilde{w}_q\}\}) \\ &= \max\{dm(\tilde{v}_1), \cdots, dm(\tilde{v}_p), dm(x_i)\} \\ &\qquad + dm(x_\ell) \max\{dm(\tilde{w}_1), \cdots, dm(\tilde{w}_q)\} \end{aligned}$$

$dm(x_\ell) = m$ であるから

$$\begin{aligned} d_{m+1}(x_i) &\leq \max\{dm(\tilde{v}_1), \cdots, dm(\tilde{v}_p), dm(x_i)\} \\ &\qquad + (m+1) \max\{dm(\tilde{w}_1), \cdots, dm(\tilde{w}_q)\} \\ &\leq d_{m+1}(t_i) \end{aligned}$$

よって (5.4) が成立する。$\Gamma(Q)$ が (5.2) を満すことは明らかである。

§6 Loop Program on RAM

これまでは基本演算として $+1$, -1 しか許されなかった。ここでは これまでの議論を加減算を許したものに

拡張しよう。

<u>Def.</u> atamic statement として <u>RAM I の loop program</u> と呼ぶ。

$$i = j + k \mid i \leftarrow j - k \mid x \leftarrow y - z \mid$$
$$x \leftarrow x_1 + x_2 + \cdots + x_n$$

ただし i, j, k は variable, x, y, z は control variable $x_1, \cdots, x_n$ $(n \geq 1)$ は control variable 又は constant である。 $x \leftarrow x_1 + \cdots + x_n$ の形の statement を <u>key statement</u> という。 P に現われる key statement の個数を P の width といい width(P) で示す。

本節では、単に loop program と言えば、RAM I の loop program のことを言う。

<u>Def.</u> P を loop program とする。このとき $x \mathrel{\overline{>}_P} y$ が成立するのは次の場合である。

(i) P が loop x do Q end の形の statement を含み Q が $y \leftarrow \alpha$ の形の key statement を含む。

(ii) x と y が異なる変数で、 P が $y \leftarrow \alpha$ の形の key statement を含み、 α が x を含む。

loop program P が simple であるとは $x_1 \mathrel{\overline{>}_P} x_2 \mathrel{\overline{>}_P} \cdots \mathrel{\overline{>}_P} x_n$, $n > 1$

なる列が存在しないときをいう。

<u>Def.</u> $g(k, n)$ を次で定義する.

$$g(0, n) = n$$

$$g(k+1, n) = 2^{g(k, n)}$$

即ち $g(k, n)$ は n の k 重の指数関数である。 各 k に対し, $\mathcal{E}^k_{xp}$ である多項式 P が存在し, $g(k, p(n))$ 時間で認識できる言語のクラスを表わす。($\mathcal{E}^k_{xp}$ は TM, RAM, SM のいずれの模型上で考えても同じクラスが定義されることに注意する.) したがって $\mathcal{P} = \mathcal{E}^0_{xp}$ であり, $\bigcup \mathcal{E}^k_{xp}$ は初等的問題全体からなるクラスである.

以下で, クラス $\mathcal{E}^k_{xp}$ が width によって特長づけられることを示す。 特に言語 L が初等的である必要十分条件は L が simple loop program on RAM で accept されることである。 証明の方針は §3, §4 と同様である.

<u>Def.</u> reduced form, normal form を次のように定義する. ここで x は control variable, $x_1, \cdots, x_n, n \geq 0$ は control variable 又は constant, i は variable.

$$\langle \text{reduced form} \rangle ::= x \leftarrow x + x_1 + \cdots + x_n \mid i \leftarrow 0 \mid$$
$$\langle \text{reduced form} \rangle ; \langle \text{reduced form} \rangle \mid$$

loop x do <reduced form> end

<type 1> ::= <key statement> | loop x do <type 1> end

<normal form> ::= <type 1> | <normal form> ; <normal form>

<u>Lemma 6.1</u> 任意の loop program P on RAM に対し、reduced な $\bar{P}$ が存在し次を満す。

(i) width(P) = width $\bar{P}$

(ii) $\mathrm{time}_P(d) \le \mathrm{time}_{\bar{P}}(d)$ for all $d \in D$

<u>Proof.</u> P の各 substatement Q に対し、Q の reduced form $\bar{Q}$ を次のように定める。i を P に現われない control variable とする。Q が key statement $x \leftarrow \ell$ なら $\bar{Q}$ は $x \leftarrow x+\ell$、Q が key statement 以外の atomic statement なら $\bar{Q}$ は $i \leftarrow 0$。Q が $Q_1 ; Q_2$ か 又は if t then Q_1 else Q_2 なら $\bar{Q}$ は $\bar{Q}_1 ; \bar{Q}_2$。Q が loop x do Q_1 end なら $\bar{Q}$ は loop x do $\bar{Q}_1$ end。すると $\bar{P}$ は lemma を満す。

Theorem 3.6 と同様にして次が示される。

<u>Theorem 6.2</u> P を reduced な simple loop program とする。このとき simple loop program Q of normal form

て次を満たすものが存在する．

(i) $\mathrm{width}(P) = \mathrm{width}(Q)$

(ii) $\mathrm{time}_P(d) \le \mathrm{time}_Q(d)$ for all $d \in D$

<u>Theorem 6.3</u> Pを simple loop program on RAM，$k = \mathrm{width}(P)$とする．$d \in D$，$n = \max_{x \in X} d(x)$とする．するとる多項式$p(n)$が存在し

$\mathrm{time}_P(d) \le g(k, p(n))$．

<u>Proof</u>. Lemma 6.1 と Theorem 6.2 より，Pは simple loop program of normal form としてよい．

$P = P_1 ; P_2 ; \cdots ; P_m$

とする．ただし各P_iは type 1，mに関する帰納法で，polinomial $p(n)$が存在し

(6.1) $\mathrm{time}_P(d) \le g(k, p(n))$

(6.2) $P(d)(x) \le g(k, p(n))$ for all $x \in X$.

が成立することを示す．mで成立したとし

$P' = P ; P_{m+1}$

を考える．$\mathrm{width}(P_{m+1}) = 0$ なら $\mathrm{width}(P') = k$．P_{m+1}は control variable を含まないから

$P_{m+1}(d)(x) = d(x)$ for all $x \in X$．

よって (6.2) より

$$P'(d)(x) \leq g(k, p(n))$$

が成立する。また P_{m+1} のネストの深さが l なら

$$time_{P_{m+1}}(d) \leq (\max_x d(x))^{l}$$

よって

$$time_{P'}(d) \leq g(k, p(n)) + (g(k, p(n))^{l}$$

$$\leq g(k, p(n)^{l+1} + l + 1)$$

となり (6.1) が成立。P_{m+1} が

loop x_1; loop x_2, …, loop x_l do $x \leftarrow y_1 + y_2 + \cdots + y_s$ end … end end

の形のとき $n = \max_{y \in x} d(y)$, $s \leq 2^{c}$, $c \geq 1$, とすると

$$P_{m+1}(d)(x) \leq s^{n^{l}} \leq 2^{cn^{l}}$$

$$time_{P_{m+1}}(d) \leq s^{n^{l}} \leq 2^{cn^{l}}$$

よって

$$time_{P'}(d) \leq g(k, p(n)) + 2^{cg(k, p(n))^{l}}$$

$$\leq g(k+1, (c+1)(p(n)^{l+1} + l + c + 1))$$

となり、(6.1), (6.2) が成立する。

Lemma 4.1 と同様に次が示される。

<u>Lemma 6.4</u> $L \in \mathcal{E}^{k}_{xp}$ なら、L は width $(P) = k$ の Simple loop program で受理できる。

proof　P_i, R を次のように定める。

$$P_i = \mathrm{loop}\ \ x_i\ \ \mathrm{do}\ \ x_{i+1} \leftarrow x_i + x_i\ \ \mathrm{end}$$

$$R = P_1; P_2; \cdots; P_k$$

とすると　$d(x) = n$ とすると

$$R(d)(x_{k+1}) \geq g(k, n)$$

となる。　あとは Lemma 4.1 と同じ。

Corollary 6.5　$L \in \mathcal{E}^k_{xp}$ である必要十分条件は width が k の simple loop program on RAM のプログラムで L が受理されること。

§7 Conclusion

ここでは制限されたプログラムのクラスについて、その時間量の上限が syntactical に解析できることを示した。同様の方法で space complexity の上限も計算できる。[1]。また配列の添字変数がその配列の範囲を越えないことのチェックもできるであろう。　たとえば Algorithm 5.1 で得られる $V(P) = (t_1, \cdots, t_n)$ を使うと memory map d のもとでの P の計算の任意の時点の memory d' に対し

$$d'(x_i) \leq d(t_i)$$

が成立する。　いいかえれば変数 x_i の内容は計算の途中

で値 $\alpha(x_i)$ を越えない。

ここで得られた理論を現実の programming language に適用するにはさらにいくつかの考察が必要であろう。たとえば while q do α end といった while statement は loop statement と組み合わせ

loop t while q do α end

といったように使うように制限をつけなければならない。ここで t は このループをまわる回数の上限を表わす term である。syntax をゆるめると同時に、simpleness の条件も拡張する必要がある。たとえば §6 でみたように RAM 上の loop program では simpleness の条件だけでは不十分である。

文献

[1] Adachi, A., A relation between space complexity and the largest number (未刊論文)

[2] Borodin, A., Computational complexity: theory and practice, in: Currents in the Theory of Computing, A.V. Aho, ed., Prentice-Hall, Englewood Cliffs, N.J., 1973. 35-89. 守屋悦朗訳「最近の計算機理論」近代科学社

[3] Constable, R.L., On the size of programs in subrecursive hierarchies Proc. 2nd Annual ACM Symposium on Theory of Computing (1970) 1-9

[4] Constable, R.L., and A. Borodin, On the efficiency of programs in subrecursive formalisms. IEEE Conference Record 11th Annual Symposium on Switching and Automata Theory (1970) 60-67

[5] Cook, S.A., and R.A. Reckhow, Time bounded random access machines, J. Comput. Syst Sci. 7 (1973), 354-375

[6] Jones, N.D., and S.S. Muchnick, Even simple programs are hard to analyze, J. ACM 24 (1977) 338-350

[7] Kasai, T., Computational complexity of multitape Turing machines and random access machines, Publ. RIMS, Kyoto Univ. 13 (1977), 469-496

[8] Kasai, T., P=NP question on program schemata, Proc. Second IBM Symp. on Mathematical Fundations of Computer Science, IBM Japan, 1977

[9] Meyer, A.R., and D.M. Ritchie, The complexity of loop programs. Proc. 22nd National ACM Conference, (1967) 465-470

[10] Grzegorczyk, A., Some class of recursive functions, Rozprawy Mathematcyzne, No. 4 Instytut Matematyczny Polskiej Akademie Nauk, Warsaw, Poland 1953 pp. 1-45