数理解析研究所講究録 1217 巻 2001 年 61-74

RECURSION SCHEMATA FOR SLOW GROWING DEPTH CIRCUIT CLASSES

SATORU KURODA
GUNMA PREFECTURAL WOMEN'S UNIVERSITY,
1395-1, KAMINOTE, TAMAMURA-MACHI, SAWAGUN, 370-1193, JAPAN

黒田覚
群馬県立女子大学

ABSTRACT. In this note we characterize iterated log depth circuit classes between AC^0 and AC^1 by Cobham-like bounded recursion schemata. We also give alternative characterizations which utilizes the safe recursion method developed by Bellantoni and Cook.

1. INTRODUCTION

The search for recursion theoretic characterizations of various complexity classes was began by A. Cobham [Cob], who characterized the class of polynomial time computable functions by a scheme now called *bounded recursion on notation.* (See also [Ro] for the proof.) The essence of this recursion scheme is two fold: firstly, on input x the recursive call is made for $|x|^{O(1)}$ times where $|x|$ is the length of x, and secondly, the growth rate is bounded by a previously defined polynomial time function.

The second condition is crucial for the characterization of resource bounded computations since the computation on each recursive call takes the value of the function as an argument, so the number of steps that each recursive call spend is, in general, proportional to the value.

In the spirit of Cobham, similar characterizations were obtained for other complexity classes such as $\mathcal{F}_{LOGSPACE}$, NC^i or AC^i. (See Lind [Li], Allen [Al], Clote [Cl88].)

It is known that without the restriction on the bound for functions generated by recursion, even much weaker scheme than bounded recursion on notation can produce all primitive recursive functions.

On the other hand, Bellantoni and Cook [BC] succeeded in eliminating this growth bound in the recursion scheme to define polynomial time functions. They used safe/normal method which originates in Leivant's tiered recursion [Le]. Such safe characterizations are further obtained for NC and NC^1 by Bloch [Bl].

This research was partially supported by Japan Society for the Promotion of Science, Grant-in-Aid for Encouragement of Young Scientists.
E-mail:satoru@gpwu.ac.jp.

The intuition for safe recursion used in [BC] is that it prohibits to replace parameters used for a recursive definition by "impredicative" values since such values may grow superpolynomially.

In this note we give characterizations of circuit classes LD^i and END^i, the classes of functions computable by a family of circuits with polynomial size and $O\left(\log^{(i)} n\right)$ depth with unbounded and bounded fan-in respectively, where $\log^{(i)} n$ is defined by,

$$\log^{(1)} n = \log n,$$
$$\log^{(i+1)} n = \log\left(\log^{(i)} n\right).$$

For $i \geq 2$ these classes lie between AC^0 and AC^1, but even inclusion (whether proper or not) between LD^i or ND^i and the classes such as NC^1, L or NL are unknown.

The main motivation of defining these classes is its relation to weak formal systems of arithmetic. Namely, in [Ku] the author characterized the class LD^i as provably total functions of the theory which is axiomatized by the weak length induction scheme:

$$\varphi(0) \wedge \forall x(\varphi(x) \to \varphi(x+1)) \to \forall x \varphi(|x|_i).$$

Another interesting feature of these classes is that although for $i = 1$, $\mathrm{LD}^1 = \mathrm{END}^1$ since both classes are equal to the class NC, it is also an open problem whether $\mathrm{LD}^i = \mathrm{END}^i$ for $i \geq 2$. Hence the LD/END hierarchy may have a different structure compared to NC/AC hierarchy.

Thus the traditional approaches to the investigations of fine structures inside logarithmic depth is those for the hierarchy

$$AC^0 \subset AC^0[2] \subseteq TC^0 \subseteq L \subseteq NL \subseteq NC^1 \subseteq AC^1,$$

and our approach gives a very different insight into this structure, which is based on the depth of circuits.

This paper is organized as follows. In section 2 we briefly overview basic concepts and notations of Boolean circuits and recursion theoretic approaches.

In section 3 we give a safe characterization of LD^i. Cobham-like characterization of LD^i is already given by the author [Ku], and the proof here utilizes that result.

In section 4 we give characterizations of LD^i. We made the definition of END^i slightly complicated in order that it includes the class AC^0 which is regarded as the smallest reasonable circuit class. This makes our recursion theoretic characterization also complicated. Namely, we cannot use the nested application of two kinds of recursion schemata.

Finally in section 5, we propose further research which links circuit complexity and proof complexity of propositional proof systems.

2. Preliminaries

2.1. Basic Notations. Throughout the paper we will consider functions over natural numbers, though numbers are often identified with its binary expansion and vice versa. The set of bit strings over alphabet $\{0,1\}$ is denoted by $\{0,1\}^*$, while ω denotes the set of natural numbers. For $x \in \omega$, $|x|$ denotes its length in binary. Furthermore, we often use the function $|x|_i$ for $i \geq 1$ which is defined by

$$|x|_1 = |x|,$$
$$|x|_{i+1} = ||x||_i .$$

A function algebra is a closure of a finite set of basic functions over additional operations which produces new functions from already given ones. As noted in the introduction similar classes are defined for other complexity classes and an excellent survey can be found in Clote [Cl].

The following is the set of basic functions which we use throughout the paper.

Definition 1. *$BASE_1$ is the following set of functions:*

- *0-ary constant:* 0,
- *projection:* $P_j^n(x_1, \dots, x_n) = x_j$,
- *successors:* $s_0(x) = 2x$, $s_1(x) = 2x + 1$,
- *bit operating functions:* $|x| = \lceil \log_2(x+1) \rceil$, $Bit(x,i) = \lfloor x/2^i \rfloor \mod 2$, $x \# y = 2^{|x| \cdot |y|}$, $Msp(x,y) = \lfloor x/2^{|y|} \rfloor$.

2.2. Iterated log depth circuits. We will define a hierarchy of circuit classes LD^i and END^i which lies between the well-known classes AC^0 and AC^1. Here we will briefly review basic notions and results on circuit classes. For details on this subject, readers are encouraged to refer textbooks of circuit complexity such as Håstad [Ha].

As usual, a *circuit* is a directed acyclic graph whose nodes are labeled by either one of logical gates $\wedge$, $\vee$, $\neg$, or an input variable. For $\wedge$-gates and $\vee$-gates, their fan-in will be given by subscript as $\wedge_n$ or $\vee_n$ respectively. However we will omit the subscript if it is clear from the context. For a family of circuit $\{C_n\}_{n \in \omega}$, its *base* is the set of logical gates used to construct each C_n. We will consider multi-output circuits so that circuits compute a finite function

$$f : \{0,1\}^m \to \{0,1\}^n,$$

where the numbers of inputs and outputs are m and n respectively. In the following we only consider uniform circuits in the following strict sense:

Definition 2. *Let $\{C_n\}_{n \in \omega}$ be a circuit family. The Direct Connection Language (DCL) of $\{C_n\}_{n \in \omega}$ is the set*

$$\{(a, b, l, 0^n) : a \textit{ is the parent of } b \textit{ in } C_n \textit{ and } l \textit{ is the label of } a\}.$$

$\{C_n\}_{n \in \omega}$ is U_{E^*}-uniform if its DCL is in DLOGTIME.

Intuitively, a circuit family is U_{E^*}-uniform if its "relational graph" which expresses the connection of gates and edges is computable in DLOGTIME.

Now we can state the definition of the class LD^i:

Definition 3. *For $i \geq 1$, LD^i is the class of functions which are computable by $n^{O(1)}$ size, $\left(\log^{(i)} n\right)^{O(1)}$ depth circuits over the base $\{(\wedge_n)_{n\in\omega}, (\vee_n)_{n\in\omega}, \neg\}$*

It is well-known that unbounded fan-in circuits have an alternative characterization by parallel random access machine called CRAM. By a CRAM we mean a random access machine which have polynomial number of processors each connected to a global memory. Let $CRAM[t(n)]$ be the class of functions computable by a CRAM in time $t(n)$. Then the following fact is well-known:

Theorem 1. *For all polynomially bounded and first order constructible $t(n)$,*

$$CRAM[t(n)] = AC[t(n)].$$

Corollary 1. *For $i \geq 1$, $LD^i = CRAM[(\log^{(i)} n)^{O(1)}]$.*

We can also define the class of $\log^{(i)} n$ depth computable classes in bounded fan-in circuits. However, the analogous definition yields a class which might not be included in AC^0. It is widely believed that AC^0 is the smallest reasonable class in circuit complexity. Hence we will strengthen the definition of our bounded fan-in circuits so as to include all AC^0 functions.

Definition 4. *For $i \geq 1$. ND^i is the class of functions which are computable by $n^{O(1)}$ size, $(\log^{(i)} n)^{O(1)}$ depth circuits over the base $\{\wedge_2, \vee_2, \neg\}$.*

For a circuit C the i-th level of C is the set of gates that are placed in the depth i of C.

Definition 5. *A circuit family is END^i if there exists a constant c such that the n-th circuit in the family has size $n^{O(1)}$, depth $\left(\log^{(i)} n\right)^{O(1)}$, and at most c levels of arbitrary fan-in AND and OR gates, the remaining levels being built from $\wedge_2$, $\vee_2$ and $\neg$ gates.*

The following inclusions are immediate from definitions above:

Proposition 1. *($i \geq 2$) $AC^0 \subseteq LD^i \subseteq AC^1$. The same also holds if we replace LD^i by END^i.*

On the other hand, no inclusions are known between our classes LD^i or END^i and the other complexity classes which lies between AC^0 and AC^1 (e.g. TC^0, NC^1, L, NL). However these classes cannot be included in any of LD^i and END^i for $i \geq 2$ from the following well-known result:

Theorem 2 (Håstad [Ha]). *Polynomial size parity circuits must have depth at least $\frac{\log n}{c+\log\log n}$ for some constant c.*

Corollary 2. *The parity function cannot be computed by LD^i circuits for $i \geq 2$.*

However we can say much more, that is, the LD hierarchy does not collapse. For a function k, define $STCONN[k(n)]$, distance k connectivity, to be the problem of given an unweighted graph G with n vertices and two fixed vertices s and t of G, determine whether or not G contains a path of length at most $k(n)$ from s to t. Beame, Impagliazzo and Pitassi proved a depth lower bound for $STCONN[k(n)]$;

Theorem 3 (Beame, Impagliazzo and Pitassi [BIP]). *For any $k(n) \leq \log^{(O(1))} n$, any polynomial-size unbounded fan-in circuit that computes $STCONN[k(n)]$ requires depth $\Omega(\log\log k(n))$.*

Thus by putting $k(n) = \log^{(i)} n$, it follows that $STCONN[\log^{(i)} n]$ cannot be in LD^{i+2}. On the other hand the following algorithm computes $STCONN[\log^{(i)} n]$ in $\log^{(i)} n$ depth:

```
begin
  input $G, s, t$;
  $c := s$;
  for $i \leq k(n)$ do in parallel
    if $c = t$ then accept and halt
    else $c :=$ adjacent node to the current one
  endfor
end)
```

So we have

Corollary 3. *For $i \geq 1$ $LD^{i+2} \subsetneq LD^i$.*

2.3. **Safe recursion.** First we shall give basic definitions. In the context of safe recursion we shall use a different notion of functions. In such cases, parameters are separated by a semicolon as $f(x_1, \ldots, x_m; y_1, \ldots, y_n)$. Parameters in the left side of the semicolon are called *normal*, while the others are called *safe*. Let NORMAL be the set of functions which have no safe parameters.

Definition 6. *$BASE_2$ is the finite set of functions which consists of the following:*

- *0-ary constant:* 0,
- *projection:*

$$P_j^{m,n}(x_1, \ldots, x_m; y_1, \ldots, y_n) = \begin{cases} x_j & \text{if } 1 \leq j \leq m, \\ y_{j-m} & \text{if } m+1 \leq j \leq m+n. \end{cases}$$

- *successors:* $S_0(; a) = 2 \cdot a$, $S_1(; a) = 2 \cdot a + 1$,
- *binary predecessor:* $P(; a) = \lfloor a/2 \rfloor$,
- *conditional:*

$$C(; a, b, c) = \begin{cases} b & \text{if } a \mod 2 = 0, \\ c & \text{otherwise.} \end{cases}$$

- *length in binary:* $L(;x) = |x|$.
- *bit function:* $BIT(;x,y) = \lfloor x/2^{|y|} \rfloor$.
- *smash function:* $\#(;x,y) = 2^{|x|\cdot|y|}$.

Note that we include base more functions than those in Bellantoni and Cook [BC] since we use weaker recursion.

Definition 7. *A function $f$ if defined by safe composition (SCOMP for short) from $g$, $u_1, \ldots, u_m$, $v_1, \ldots, v_n$ if*

$$f(\vec{x};\vec{a}) = g(u_1(\vec{x};), \ldots, u_m(\vec{x};); v_1(\vec{x};\vec{a}), \ldots, v_n(\vec{x};\vec{a})).$$

3. Characterizing the class LD^i

3.1. Weak length recursion and LD^i. First we characterize the class LD^i using normal form of weak recursion and then we go on to show a safe characterization in the next subsection.

Definition 8. 1. *A function $f$ is defined by safe concatenation recursion on notation (SCRN for short) from g, h_0 and h_1 if*

$$\begin{aligned} f(0,\vec{y};\vec{a}) &= g(\vec{y};\vec{a}) \\ f(2x,\vec{y};\vec{a}) &= s_{h_0(x,\vec{y};\vec{a})}(;f(x,\vec{y};\vec{a})), \text{ if } x \neq 0, \\ f(2x+1,\vec{y};\vec{a}) &= s_{h_0(x,\vec{y};\vec{a})}(;f(x,\vec{y};\vec{a})), \end{aligned}$$

provided that h_0, $h_1 \leq 1$.

2. *Let $i \in \omega$. A function $f$ is defined by i-Weak Bounded Recursion on Notation (i-WBRN) from $g$, h_0, h_1 and $k$ if*

$$\begin{aligned} F(0,\vec{y}) &= g(\vec{y}), \\ F(s_0(x),\vec{y}) &= h_0(x,\vec{y},f(x,\vec{y})), \text{ if } x \neq 0, \\ F(s_1(x),\vec{y}) &= h_1(x,\vec{y},f(x,\vec{y})), \\ f(x,\vec{y}) &= F(|x|_i,\vec{y}), \end{aligned}$$

provided that $F(x,\vec{y}) \leq k(x,\vec{y})$ for all $x, \vec{y}$.

Theorem 4. *U_{E^*}-uniform* LD^i *is the smallest class of functions containing $BASE_1$ and closed under composition, CRN and i-WBRN operations.*

Proof. Let $K$ be the closure of $BASE_1$ under composition, CRN and $i$-WBRN. To show that $K \subseteq LD^i$, it suffices to show that LD^i is closed under $i$-WBRN since other cases are identical to the proof of Clote and Takeuti's result stating that AC^0 is the closure of $BASE_1$ under composition and CRN. We shall show that $CRAM[\log^{(i)} n]$ is closed under $i$-WBRN. Let $f$ be defined by $i$-WBRN from $g$, h_0, h_1 and $k$ which are computable by some CRAM's in time $(\log^{(i)} n)^{l_g}$, $(\log^{(i)} n)^{l_{g_0}}$, $(\log^{(i)} n)^{l_{h_1}}$ and $(\log^{(i)} n)^{l_k}$, respectively. On input $x$, the CRAM $M$ for $f$ computes as follows: in stage $t$ simulate h_0 or h_1 according to the $t$th bit of $|x|_{i+1}$ and finally simulate $g$. By the inductive hypothesis each step requires at most $(\log^{(i)} n)^l$ steps where $l = \max\{l_g, l_{h_0}, l_{h_1}\}$, so $M$ also terminates in $(\log^{(i)} n)^{l+1}$. It is also easy to see that the number of processors required by M is polynomial in $|x|$.

For the opposite direction we shall give a proof that utilizes a direct construction of LD^i circuits by weak recursion operations. Let $C = \{C_n\}_{n\in\omega}$ be an LD^i circuit family computing a function $f : \{0,1\}^* \to \{0,1\}^*$. Then C_n has size $n^{O(1)}$ and depth $\left(\log^{(i)} n\right)^k$ for some $k \in \omega$. Let $p(n)$ be the polynomial which bounds the number of gates in C_n. The proof proceeds by induction on k. For the base case let $k = 1$. Let $x \in \{0,1\}^*$ be an input bit string to C. First define

$$EncodeInput(x) = \langle (x)_1, \ldots, (x)_{|x|} \rangle$$

where $(x)_i$ denotes the i-th bit of x. Note that from the length of the input bit x we can decide which C_n is to be taken to compute $f(x)$, namely $C_{|x|}$. Since C is U_{E^*}-uniform, this choice can be made in DLOGTIME. Next we define the function $Eval_C(w,j)$ which outputs the (code of the) output string of j-th level of $C_{|x|}$ if w is a valid code of an input string to the j-th level. The construction of this function is divided into two phases.

For the first phase let g_l be the l-th gate from the left side in the j-th level of $C_{|x|}$. Then we can compute the bit position in w which is connected to g_l by some AC^0 function. That is, make a linear search on w and in i-th step of the search check whether $(w)_i$ is the input to g_l using the DCL for C. This algorithm yields the input bit string for each gate g_l. Furthermore, it is well-known that linear searches on polynomial length sequences can be done in AC^0.

In the second phase, evaluate each output of g_l by using that input given by the first phase above. (Note that any gate in $C_{|x|}$ is trivially regarded as an AC^0 circuit.)

Thus we can compute the function $Eval_C(w,j)$ by the following algorithm:

```
begin
  input $w, j$;
  $G := \langle\rangle$; (empty sequence)
  concatenate each gates $g_l$ in the $j$-th level of $C_{|x|}$;
  to $G$ one by one;
  $out := \langle\rangle$;
  for $i := 1$ to $length(G)$ do
    compute the input string $w_i$ for the gate $(G)_i$;
    evaluate $(G)_i$ on input $w_i$;
    concatenate the value to $out$;
  endfor
end
```

By the preceding argument, this algorithm can be implemented by an AC^0 circuit.

Now, starting from *EncodeInput*(x) and iterating $\log^{(i+1)} n$ times the evaluation of the function $Eval_C$, we obtain the output of C_n on input x. This iteration procedure can be expressed by i-WBRN. operation since each level of output cannot exceed $p(n)$ and hence the bounding term of i-WBRN is of the form $|t^{p(n)}|$ for some term t. Namely f can be defined by i-WBRN as follows:

$$\begin{aligned}
&F(0, x) = EncodeInput(x),\\
&F(s_0(n), x) = Eval_C(n, F(n, x)), \text{ if } n \neq 0,\\
&F(s_1(n), x) = Eval_C(n, F(n, x)),\\
&f(x) = F(|x|_i, x).
\end{aligned}$$

If $k \geq 2$, then by the induction hypothesis depth $(\log^{(i)} n)^{k-1}$ sub-circuits of C_n can be evaluated by functions in K. Furthermore, gathering these outputs can be done by some AC^0 function. So applying i-WBRN one more time yields the output of C_n. □

3.2. **Safe recursion for** LD^i. The main idea of our characterization of the class LD^i is that the number of recursive calls which are made by the scheme of recursion on notation corresponds to the depth of circuits. So $\log^{(i)} n$ depth corresponds to $|x|_i$ many recursive calls. To formalize the argument, we first define the following function:

Definition 9. *The function $H_i(x)$ for $i \geq 1$ is defined as follows:*

$$\begin{aligned}
&H_0(x) = \lfloor x/2 \rfloor,\\
&H_{i+1}(x) = 2^{H_i(|x|_i)}
\end{aligned}$$

Note that by $|x|_i$ many iteration of H_i to x reaches 0. So our weak safe recursion is as follows:

Definition 10. *A function f is defined by safe i-weak recursion on notation (i-SWRN for short) from g and h if*

$$\begin{aligned}
f(0, \vec{y}; \vec{a}) &= g(\vec{y}; \vec{a}),\\
f(x, \vec{y}; \vec{a}) &= h(x, \vec{y}; \vec{a}, f(H_i(x), \vec{y}; \vec{a})), \text{ if } x \neq 0.
\end{aligned}$$

We also need the CRN operation which can be stated in the context of safe recursion as follows:

Definition 11. *A function f is defined by safe concatenation recursion on notation (SCRN for short) from g, h_0 and h_1 if*

$$\begin{aligned}
f(0, \vec{y}; \vec{a}) &= g(\vec{y}; \vec{a})\\
f(2x, \vec{y}; \vec{a}) &= s_{h_0(x,\vec{y};\vec{a})}(; f(x, \vec{y}; \vec{a})), \text{ if } x \neq 0,\\
f(2x+1, \vec{y}; \vec{a}) &= s_{h_1(x,\vec{y};\vec{a})}(; f(x, \vec{y}; \vec{a})),
\end{aligned}$$

provided that h_0, $h_1 \leq 1$.

Note that we let any recurrence take place on normal parameters.

In this section we show that i-SWBRN captures the class LD^i.

Theorem 5. *Let B_i be the smallest class of functions containing $BASE_2$ and closed under SCOMP, SCRN and i-SWBRN operations. Then a function f is in* LD^i *if and only if $f(\vec{x};$ is in B_i.*

This theorem can be proved by almost the same idea as in Bellantoni and Cook [BC]. Namely, Lemma 4.2 in [BC] can be modified as

Lemma 1. *If $f \in \mathrm{LD}^i$ then there exists a function $f' \in B_i$ and a monotone increasing polynomial p_f such that $f(\vec{x}) = f'(w; \vec{x})$ for all $w \geq p_f(|\vec{x}|)$.*

The proof in [BC] can be directly applied to Lemma 1 since we let B_i include bit operating functions such as $\#$ and BIT. Thus from Lemma 1 we obtain

Theorem 6. *If $f(\vec{x}) \in \mathrm{LD}^i$ then $f(\vec{x};) \in B_i$.*

The opposite inclusion also follows from easy induction. We omit the detail here, so reader should refer to [BC].

4. Characterizing the class END^i

4.1. Divide and Conquer for END^i.

The characterization of END^i is a little more complicated than those for other classes such as $\mathcal{F}_{PTIME}$ or LD^i.

First, we slightly change base functions so that they can be computed by NC^0 circuits. It is known that as multi-output circuits, the computational power of NC^0 is fairly strong. Namely, it is proved in Bloch [Bl] that the following functions can be computed in NC^0.

Definition 12. *$BASE_3$ is the following set of functions:*

- *constant:* 0,
- *projection:* $P_j^n(x_1, \cdots, x_n) = x_j$,
- *conditional:*

$$C(;a,b,c) = \begin{cases} b & \textit{if } a \mod 2 = 0, \\ c & \textit{otherwise.} \end{cases}$$

- *bit operations:*
  - $Msp(x,y) = \lfloor x/2^{|y|} \rfloor$,
  - $Conc(x,y) = x \cdot 2^{|y|} + y$,
  - $Bh(x) = x \mod 2^{\lceil |x|/2 \rceil}$, $Fh(x) = Msp(x, Bh(x))$,
  - $Ins_i(x) = x$ *with a i inserted after each bit for* $i = 0, 1$.
- *logical operations:*
  - $Not(x) =$ *the one's complement of* x,
  - $Or(x,y) =$ *the bitwise OR of* x *and* y.

The number of these base functions may be reduced, however, we do not go into the subject any further.

The new recursion we use is the following:

Definition 13. *A function f is defined by i-Divide and Conquer Recursion from g, h and k if*

$$\begin{aligned} F(0,\vec{y}) &= g(\vec{y}), \\ F(x,\vec{y}) &= h(x,\vec{y},F(Fh(x),\vec{y}),F(Bh(x),\vec{y})), \text{ for } x \neq 0, \\ f(x,\vec{y}) &= F(|x|_i,\vec{y}), \end{aligned}$$

provided that $F(x,\vec{y}) \leq k(x,\vec{y})$ for all x and $\vec{y}$.

The difficulty with the characterization of END^i is that we cannot show that this class is closed under i-DCR. This is illustrated as follows. Suppose END^i is closed under i-DCR. Intuitively, this means that the $\log^{(i)} n$ iteration of END^i circuits is also an END^i circuit. However this might not be the case since each END^i circuits in the iteration contains a constant number of unbounded fan-in levels and thus $\log^{(i)} n$ iteration of such circuits yields $\left(\log^{(i)} n\right)^{O(1)}$ levels in the circuits obtained. So the intuition of Theorem 7 is that in the construction of an END^i circuit, we divide the generation of subcircuits in two phases: first construct $\left(\log^{(i)} n\right)^{O(1)}$ depth fan-in 2 components using BASIC functions and i-DCR. Then taking the closure of the set of such components by composition and CRN operations yields END^i circuits.

First we give a function algebra based on Cobham-like bounded recursion scheme and then simulate it by weak safe recursion as we did in the previous section.

We begin by showing that END^i is closed under CRN operation. That is,

Proposition 2. *Suppose f is defined by CRN from g, h_0 and h_1 each in* END^i. *Then* $f \in \mathrm{END}^i$.

Definition 14. *Let N_i be the closure of $BASE_3$ under COMP and i-DCR operations and N_i^* be the class of functions containing N_i and closed under COMP and CRN operations.*

Note that we prohibit i-DCR and CRN to nest alternately due to the reason we already stated above. Now the main theorem in this section is the following:

Theorem 7. END^i *is the class of functions in N_i^*.*

First we will show the inclusion from left to right:

Lemma 2. *Any function in N_i can be computed by an* ND^i *circuit family.*

Proof. By induction on the construction of $f \in N_i$. It is proved in Bloch [Bl] that all BASE functions are NC^0 computable.

If $f(x) = g(h(x))$ and g and h are computable by some ND^i circuits C_g and C_h respectively, then by plugging in the outputs of H to inputs of G we obtain a circuit which computes f.

Suppose f is definable by i-DCR from g and h. Let T_f be a complete binary tree of depth $|x|_i$ for input x and replace each internal node of the binary tree by C_h and leaf by C_g will obtain an ND^i circuit C_f which computes f. □

Lemma 3. *Let $f \in N_i^*$. Then there exists a family of* END^i *circuits which computes f.*

Proof. By induction on the construction of $f \in EN_i^*$. The base cases are already proved as Lemma 2 and the case for CRN is also straightforward. Hence we shall show that END^i is closed under COMP operations as $f(\vec{x}) = g(h(\vec{x}))$. (For simplicity we assume that g has only one parameter and the general case is shown analogously.) By inductive hypothesis g and h have END^i circuits C_g and C_h respectively. Let C_f be the circuit which is obtained by plugging the outputs of C_h to the inputs to C_g. It is easy to see that C_f satisfies the size and depth requirements. Furthermore, as C_g and C_h contains only constantly many unbounded fan-in gates in each computation path, so does C_f. □

For the opposite inclusion we shall give a direct construction of functions which simulates END^i circuits. First we will sketch the idea of the proof. The idea is that each END^i circuit can be divided into a constant number of ND^i subcircuits $C_1, \dots, C_n$ which are separated by some levels with unbounded fan-in gates. Note that we are considering multi-output circuits, so each $C_1, \dots, C_n$ and all unbounded fan-in levels can be considered as circuits. Hence C_i's are simulated by N_i functions while unbounded fan-in levels are considered as AC^0 circuits.

Now a single concatenation of these components can be defined by a single use of composition operation, therefore a constant number of composition yields the whole circuit.

This process is formalized as follows:

Lemma 4. *If $f(\vec{x})$ is computed by some U_{E^*}-uniform* END^i *circuit family, then $f \in N_i^*$.*

Proof. It is too messy to describe the whole simulation of circuits by functions and so it might be too long and incomprehensive. So instead we give a proof in a slightly informal manner so that it can be formalized in more strict proof by a routine work.

Let C be an END^i circuit. A subcircuit C' of C is called an ND^i component if it is a ND^i circuit (i.e. contains no unbounded fan-in gates) whose input and output gates are connected to unbounded fan-in gates. In other words, an ND^i component is a maximal ND^i subcircuit of C.

Given an END^i circuit family $\{C_n\}_{n\in\omega}$, it is straightforward to see that each C_n can be decomposed into a constant number of ND^i components. Furthermore, without loss of generality, we can assume that all C_n $(n \in \omega)$ has same number of ND^i components by adding "dummy" components to

C_n if necessary. Also note that this construction can be done without losing the uniformity condition.

Now the proof proceeds by induction on the number of ND^i components k in $\{C_n\}_{n\in\omega}$.

If $k = 1$ then the circuit is either an ND^i circuit or an ND^i circuit attached to unbounded fan-in gates on both input and output gates.

Suppose first that it is an ND^i circuit. Let us construct a function computing the j-th bit of C_n on input $x \in \{0,1\}^*$. Then combining all output bits into a natural number can be done by an AC^0 operation, which trivially can be manipulated by the algebra N_i^*.

Note that an ND^i circuit can be regarded as a divide and conquer strategy which is executed in $\left(\log^{(i)} n\right)$ depth. Hence i-DCR operation can generate a function computing the j-th bit of the output since, as we consider U_{E^*}-uniform circuits, each gate can be recognized by a DLOGTIME algorithm, which is also available in END^i. The detail of the construction is routine and is left to the reader.

For the second case, first construct N_i^* function f_{in} and f_{out} computing the unbounded fan-in gates attached on the input and output sides respectively. Let g be the N_i^* function computing the intervening ND^i component which is constructed as the above procedure. Then the function computing the output of the whole circuit is the composition of these functions

$$F(x) = f_{out}(g(f_{in}(\vec{x}))).$$

For the induction step, consider a circuit family whose number of ND^i components are $k+1$. Divide the circuit into the upper most ND^i component C_1 and the rest of the circuit C_2. Then similar to the base case we can construct an N_i^* function f_1 computing C_1 and as C_2 contains k components, the inductive hypothesis yields a function f_2 computing C_2. Now the whole circuit can be computed by the composition of these functions $f_1(f_2(\vec{x}))$. $\square$

4.2. Safe recursion for END^i. Finally we shall present a safe characterization of END^i. This is straightforward as we know how the bounded recursion scheme can be transformed into safe one.

Definition 15. *A function f is defined by safe i-Divide and Conquer Recursion (i-SDCR) from g and h if*

$$\begin{aligned}
F(0,\vec{y};\vec{a}) &= g(\vec{y};\vec{a}),\\
F(x,\vec{y};\vec{a}) &= h(x,\vec{y};\vec{a},F(Fh(x),\vec{y};\vec{a}),F(Bh(x),\vec{y};\vec{a})),\\
f(x,\vec{y};\vec{a}) &= F(|x|_i,\vec{y},\vec{a}).
\end{aligned}$$

Definition 16. *Let SN_i be the closure of $BASE_3$ under SCOMP and i-SDCR operations and SN_i^* be the class of functions containing SN_i and closed under SCOMP and SCRN operations.*

Using the same method as in section 3, we can show the following:

Theorem 8. *Let f be a function. Then $f(\vec{x}) \in N_i^*$ if and only if $f(\vec{x};) \in SN_i^*$.*

5. Concluding Remarks

In this note we characterize in a machine-independent manner those classes of functions which have slow growing depth circuits. A major application of them is its connection to propositional proof complexity. For example Cook [Co] used Cobham's result to define an equational system PV which simulates all polynomial length reasoning by introducing all polynomial time functions in the system. PV corresponds to extended Frege system, and afterward, Clote [Cl93], Arai [Ar], and Pitt [Pi] defined similar systems which simulate polynomial size Frege proofs. It is known that the computational complexity needed for these simulations is ALOGTIME.

Hence a function algebra, together with a logical system, links complexity classes to propositional proof systems. Thus in this line of investigations, it is natural to ask what proof system corresponds to the classes LD^i and END^i. Finding a bounded arithmetic theory which corresponds to END^i is also plausible.

References

[Al] Allen, W.: Arithmetizing Uniform NC. Annals of Pure and Applied Logic. **53** (1991) 1–50.

[Ar] Arai, T.: Bounded Arithmetic AID for Frege System. preprint. (1998). Available via http://www.fields.utoronto.ca/preprints/FI/

[BIP] Beam, P., R. Impagliazzo, T. Pitassi. Improved depth lower bounds for small distance connectivity. In: Proc. of 36th IEEE Symposium on Foundations of Computer Science. (1995) 692–703.

[BC] Bellantoni, S., S. Cook.: A New Recursion-theoretic Characterization of Polytime Functions. Computational Complexity. **2** (1992), 97–110.

[Bl] Bloch, S.: Function-algebraic Characterizations of Log and Polylog Parallel Time. Computational Complexity 4(2) (1994) 175–205.

[Cl88] Clote, P.: Sequential, Machine-independent Characterizations of the Parallel Complexity Classes ALOGTIME, AC^k, NC^k and NC. In: Feasible Mathematics. Birkhäuser, (1989) 49–69.

[Cl93] Clote, P.: Polynomial Size Frege Proofs of Certain Combinatorial Principles. In: Arithmetic, Proof Theory and Computational Complexity. Oxford Univ. Press (1993) 730–117.

[Cl] Clote, P.: Computation Models and Function Algebras, preprint.

[Cob] Cobham, A.: The Intrinsic Computational Difficulty of Functions. In: Logic, Methodology and Philosophy of Science II. North-Holland, (1965) 24–30.

[Co] Cook, S.: The Complexity of Theorem Proving Procedures. In: Conference Record of Third Annual ACM Symposium on Theory of Computing. (1971) 151–158.

[Ha] Håstad, J.: Computational Limitations for Small-Depth Circuits. The MIT Press, (1986)

[Ku] Kuroda, S.: Weak Length Induction and Slow Growing Depth Boolean Circuits. submitted. (1999)

[Le] Leivant, D.: A Foundational Delineation of Computational Feasibility. In: Proc. Sixth Annual IEEE Symposium on Logic in Computer Science. (1991) 2–11.

[Li] Lind, J. C.: Computing in Logarithmic Space. Technical Report Massachusetts Institute of Technology, (1974)

[Pi] Pitt, F.: A Quantifier-Free Theory Based on a String Algebra for NC^1. In:Proc. DIMACS Workshop on Feasible Arithmetics and Proof Complexity. (1998) 229–252.

[Ro] Rose, H. E.: Subrecursion: Function and Hierarchies. Oxford Logic Guide, Clarendon Press (1984)