数理解析研究所講究録
950 巻 1996 年 26-32

PRAM および対数時間一様な論理回路族に基づく計算量の階層

九州芸術工科大学 岩本 宙造 (Chuzo Iwamoto)
九州大学・工学部 岩間 一雄 (Kazuo Iwama)

1 Introduction

It is well recognized that to separate NC^k from NC^{k+1} is very difficult. It is even unknown whether all sets in P are recognized by logspace-uniform circuits of linear size and logarithmic depth. This might be the reason why we usually think, unlike the sequential case, it is hopeless to try to prove hierarchies for parallel complexities. However, it should be noted that this perception is only reasonable for just one model, ***logspace-uniform circuits***. In this paper, it is shown that (i) there exist a constant d and a language L such that L is recognizable in time $dT(n)$ by some PRIORITY CRCW PRAM but is not recognizable in time $T(n)$ by any PRIORITY CRCW PRAM if the number of processors is fixed and (ii) there exist constants c, d and a language L such that L is recognizable by some family of DLOGTIME-uniform circuits of size $(Z(n))^c$ and depth $dT(n)$ but is not recognizable by any family of DLOGTIME-uniform circuits of size $Z(n)$ and depth $T(n)$ if $T(n)$ is not bounded by $O(\log n)$. The above result (i) improves the hierarchy of PRAM-based parallel complexity classes shown by Kirchherr [19], and as for (ii), little surprisingly no such hierarchies based on circuits have been presented.

Kirchherr [19] showed that there exists a language L which is not recognizable in time $\log^i n$ by any PRIORITY CRCW PRAM with n^j processors but is recognizable in time $\log^{i+8} n$ by a PRIORITY CRCW PRAM with $n^{96j+104}$ processors. This hierarchy is obtained by transforming it into the hierarchy of time- and reversal-bounded deterministic TMs. This transformation might be the reason why his hierarchy is much less tight than our present one. In this paper, we apply the diagonalization method directly to PRAMs, by which we can show that a constant increase of parallel time (and no increase of processors) yields a new PRAM-based parallel complexity class. In the sequential case, a tight time-hierarchy theorem is known for RAMs [12].

More precisely, our second result shows: There exist constants c and d such that if $T_2(n) > dT_1(n)$ and $Z_2(n) > (Z_1(n))^c$ for all n greater than some n_0, then DLT-U$(T_1(n), Z_1(n)) \not\supseteq$ DLT-U$(T_2(n), Z_2(n))$, where DLT-U$(T(n), Z(n))$ is the class of sets recognizable by DLOGTIME-uniform circuits of depth $T(n)$ and size $Z(n)$. This immediately implies hierarchies for big-O complexities, like

$$\begin{aligned} & \text{DLT-U}(O(\log n), O(n^2)) \\ \subsetneq\ & \text{DLT-U}(O(\log n \log\log n), O(n^{2c})) \\ \subsetneq\ & \text{DLT-U}(O(\log^2 n), O(n^{2c^2})) \subsetneq \cdots \subsetneq P. \end{aligned}$$

Recall that in the case of logspace-uniform circuits, even whether LS-U$(O(\log n), O(n)) \subsetneq P$ is not known, where LS-U$(T(n), Z(n))$ is the class of sets recognizable by logspace-uniform circuits of depth $T(n)$ and size $Z(n)$.

At the same time, however, it is also a fact that DLOGTIME-uniform circuits do not seem to differ that much from logspace-uniform circuits, since if $k \geq 2$ then NC^k coincides for both uniformities [21]. One might think that a proper hierarchy under DLOGTIME-uniformity could imply a proper hierarchy under logspace uniformity, since (i) logspace-uniform circuits can be translated to DLOGTIME-uniform circuits with constant and polynomial loss in depth and size [21], (ii) constant and polynomial increase of depth and size strictly enlarges the complexity class of DLOGTIME-uniform circuits (our new result in this paper), and (iii) all DLOGTIME-uniform circuits are obviously logspace-uniform.

Nevertheless, (fairly standard) diagonalization works for DLOGTIME-uniform circuits but does not seem so for logspace-uniform ones. The main contribution of this paper is to call attention to this distinction between the two uniformities. (The answer to the above skepticism that our hierarchy might imply logspace-uniform hierarchy will be given in Section 3.)

Since we consider fan-in 2 circuits in this paper, our hierarchy theorem does not hold for depth less than $\log n$. In the class NC^1, several separation results have been known. For example, there is a non-collapsing hierarchy in AC^0, which is the class of problems solvable by constant depth, polynomial-size, unbounded fan-in circuits. It is known [22] that there are problems in $AC_k^0 - AC_{k-1}^0$ for each $k > 0$, where AC_k^0 is the class of problems solvable by DLOGTIME-uniform, depth-k, polynomial-size, unbounded fan-in circuits. Also, it is known [2, 13] that the exclusive OR function is not in AC^0, which implies that $AC^0 \subsetneq NC^1$. On the other hand, it is open whether ACC $\subsetneq$? NC^1 [4], where ACC is the class of problems solvable by constant depth, polynomial-size, unbounded fan-in Boolean circuits in which any "MOD(k)-gate," $k > 1$, may be used. ([3] conjectured that ACC $\neq NC^1$.) One of the results inside NC^1 is that there are problems complete for DLOGTIME-uniform NC^1 under DLOGTIME reductions [5, 9]. (More information on the class NC^1 may be found in [6, 7].)

On the other hand, almost no results have been known about the NC^k hierarchy. It is open whether there exists an integer k such that $NC^k = NC^{k+1}$ [17]. No one has succeeded in proving $NC^1 \neq P$ (or even ACC $\neq$ NP). Also, the NC^k hierarchy collapses if NC has complete problems under either log-space or NC^1 reductions (see [10] for the NC^1-reducibility). One approach to studying parallel complexity classes is characterizing circuits by TMs. Ruzzo [21] showed that $NC^k = \text{ASPACE,TIME}(\log n, \log^k n)$ for $k \geq 2$. If the circuits are U_B-uniform [21] (U_B-uniform circuits of depth $T(n)$ are constructed by $T(n)$-space DTMs), it is known that $\text{DTIME}(T) \subseteq \text{uniform size}(T \log T) \subseteq \text{DTIME}(T \log^3 n)$ [20] and $\text{NSPACE}(S) \subseteq \text{uniform depth}(S^2) \subseteq \text{DSPACE}(S^2)$ [8]. Also it seems quite sure that the parallel complexity of some problems gradually increases with the value of parameters. Those problems include k-connectivity [18], α-connectivity [14, 15], and some artificial language having unlimitedly lower parallel time-complexities [16].

DLOGTIME-uniformity has slightly different definitions in the literature. Our present definition is the one using the extended connection language. The same results hold for another definition using the direct connection language if $T(n) = \Omega(\log n \log\log n)$. Also, similar hierarchy holds for unbounded fan-in circuits. Note that our result needs to specify an explicit size of circuits; it does not say anything about whether $NC^k \subsetneq NC^{k+1}$, or even whether $NC^1 \subsetneq P$, under DLOGTIME-uniformity, which is still open.

2 Definitions and Results

All Turing Machines (TMs) in this paper are deterministic k-tape TMs with only 0 and 1 as their tape symbols.

Our PRAM is essentially the same model as defined in [23]. A PRAM has a *common memory*, $M[0], M[1], M[2], \ldots$, and a sequence of *processors* (RAMs) operating synchronously in parallel. (See [1] for RAM.) Each processor of the PRAM has its own *local memory*, $R[0], R[1], R[2], \ldots$, and has instructions for addition, subtraction, logical OR, AND, conditional branches based on predicates $=$ and $<$, and reading and writing into its local memory. The processors can access to the common memory, and each processor has instructions for reading from and writing into the common memory using its local memory to specify the common memory address. If more than one processor attempts to write the same location in common memory at the same time, the lowest numbered processor succeeds. All processors have the same program. The input string of length n is given in $M[0], M[1], \ldots, M[n-1]$. The computation halts when all processors have halted. The PRAM operates in time $T(n)$ if it halts within $T(n)$ steps on any input of length n. When the PRAM accepts (rejects) the input string, symbol 1 (0) appears in $M[0]$ after $T(n)$ steps. The complexity of PRAM program is measured according to the uniform cost criterion.

The definitions of circuits are mostly from [21]. A *combinatorial circuit* is a directed acyclic graph, where each node (gate) has indegree $d \leq 2$, and labeled by some Boolean function of d variables, or has indegree 0 and is labeled by "x" (an input). Nodes with outdegree 0 are *outputs*. In this paper, we consider a *family* $C = (\alpha_1, \alpha_2, \ldots, \alpha_n, \ldots)$ of circuits, where α_n has n inputs and one output. We denote the size and depth of α_n by $Z(n)$ and $T(n)$, respectively.

Let $g(p)$ denote the gate reached by following the path p of inputs to g. For example, $g(\varepsilon)$ is g, $g(L)$ is g's left input, $g(LR)$ is g's left input's right input, and so on. The *standard encoding* $\overline{\alpha}_n$ is a string of 4-tuples $\langle n, g, p, y\rangle$, where $g \in \{0,1\}^*$, $p \in \{\varepsilon, \text{L}, \text{R}\}$, and $y \in \{x, \wedge, \vee, \neg\} \cup \{0,1\}^*$ such that in α_n either (i) $p = \varepsilon$ and gate g is a y-gate, $y \in \{x, \wedge, \vee, \neg\}$, or (ii) $p \neq \varepsilon$ and gate $g(p)$ is numbered y, $y \in \{0,1\}^*$. The *direct connection language* L_{DC} of the family C is the set of strings of the form $\langle n, g, p, y\rangle$. The family of circuits, $C = (\alpha_1, \alpha_2, \ldots, \alpha_n, \ldots)$, of size $Z(n)$ and depth $T(n)$ is said to be *logspace-uniform* if the mapping $n \longrightarrow \overline{\alpha}_n$ is computable by a DTM in space $\log Z(n)$.

The definition of the *extended encoding* $\widehat{\alpha}_n$ is the same as $\overline{\alpha}_n$, except $p \in \{\text{L}, \text{R}\}^*$ and $|p| \leq \log Z(n)$. The *extended connection language* L_{EC} of the family C is the set of strings of the form $\langle n, g, p, y\rangle$. The family of circuits, $C = (\alpha_1, \alpha_2, \ldots, \alpha_n, \ldots)$, of size $Z(n)$ and depth $T(n)$ is said to be *DLOGTIME-*

uniform if there is a DTM recognizing L_{EC} which takes time $O(\log Z(n))$. This definition is the same as U_{E}-uniform in [21].

Remark. Another definition of DLOGTIME-uniformity uses the direct connection language L_{DC}, i.e., the circuit family C of size $Z(n)$ and depth $T(n)$ is said to be DLOGTIME-uniform if there is a DTM recognizing L_{DC} which takes time $O(\log Z(n))$. Theorem 2 also holds for this definition of DLOGTIME-uniformity if $T(n) = \Omega(\log n \log\log n)$.

Now we are ready to show our main results. (The proof of Theorem 1 is omitted. The proof of Theorem 2 is given in Section 3.)

Theorem 1. *Suppose $T_1(n)$ is a function which is not constant and is computable by a $T_1(n)$-time PRAM with $P(n)$ processors. Then, there exist a language L, a constant d, and an integer n_0 such that (i) $T_2(n) > dT_1(n)$ for all $n \geq n_0$ and (ii) L is recognizable by a $T_2(n)$-time PRAM with $P(n)$ processors but is not recognizable by any $T_1(n)$-time PRAM with $P(n)$ processors.*

Let PRAM$(T(n), P(n))$ be the class of sets recognizable by PRAMs with $P(n)$ processors in time $T(n)$. If $T_2(n) = dT_1(n)$, then $T_2(n) > T_1(n)$ for all integers $n > 0$. Hence:

Corollary 1. *For a similar $T_1(n)$ as above, there exists a constant d such that PRAM$(T_1(n), P(n)) \subsetneq$ PRAM$(dT_1(n), P(n))$.*

Theorem 2. *Suppose that $T_1(n)$ is a polylogarithmic function not bounded by $O(\log n)$ and $Z_1(n)$ is a polynomial function such that $\log Z_1(n)$ and $T_1(n)$ are computable by $O(\log n)$-time DTMs if input n is given in binary. Then, there exist a language L, constants c, d, and an integer n_0 such that (i) $T_2(n) > dT_1(n)$ and $Z_2(n) > (Z_1(n))^c$ for all $n \geq n_0$ and (ii) L is recognizable by a family of DLOGTIME-uniform circuits of size $Z_2(n)$ and depth $T_2(n)$ but is not recognizable by any family of DLOGTIME-uniform circuits of size $Z_1(n)$ and depth $T_1(n)$.*

The functions $T_1(n)$, computable by $O(\log n)$-time DTMs, includes many specific functions, such as $\log n \log^* n, \log n \log\log n$, and $\log^k n$.

Corollary 2. *For similar $T_1(n)$, $Z_1(n)$, and constants c, d as above, (i) DLT-U$(T_1(n), Z_1(n)) \subsetneq$ DLT-U$(dT_1(n), (Z_1(n))^c)$. (ii) If $\lim_{n\to\infty} T_1(n)/T_2(n) = \lim_{n\to\infty} (Z_1(n))^c/Z_2(n) = 0$ then DLT-U$(T_1(n), Z_1(n)) \subsetneq$ DLT-U$(T_2(n), Z_2(n))$.*

3 Proof of Theorem 2

Let β_n be circuits of size $Z_1(n)$ and depth $T_1(n)$, and α_n be circuits of size $Z_2(n)$ and depth $T_2(n)$. In order to prove that the class of languages recognizable by β_n is properly contained in the class of languages recognizable by α_n by the diagonalization method, we will show that (i) α_n can generate the extended encoding $\widehat{\beta}_n$ of any β_n and (ii) α_n can simulate β_n. Under the DLOGTIME-uniformity, the extended connection language is recognized by an $O(\log n)$-time DTM. By simulating this DTM, α_n generates the extended encoding $\widehat{\beta}_n$ of any β_n. For (ii), we can use the universal circuit U of Cook and Hoover [11], which can simulate any circuit β_n if the extended encoding $\widehat{\beta}_n$ is given to U as its input.

The depth $T_1(n)$ of α_n may be any well-behaving function not bounded by $O(\log n)$. The reason why the theorem does not hold for $T_1(n) = O(\log n)$ is that α_n must be able to simulate any $O(\log n)$-time DTM which has as many states, tapes and symbols as possible. (This is similar to the space-hierarchy theorem of DTM, i.e., the languages recognizable by $S_1(n)$-space DTMs is properly contained in those by $S_2(n)$-space DTMs for any well behaving function $S_2(n)$ not bounded by $O(S_1(n))$, but the theorem does not hold for $S_2(n) = O(S_1(n))$.) It is also shown in [11] that the extended encoding can be obtained from the standard encoding by the conversion circuit of depth $O(\log n \log\log n)$. Therefore, Theorem 2 also holds for DLOGTIME-uniformity with the direct connection language if $T(n) = \Omega(\log n \log\log n)$.

This might be a good point to give an answer to the skepticism in Section 1. To be exact, [21] says that any *single* language recognizable by a family of logspace uniform circuits of size $Z(n)$ and depth $T(n)$ is recognizable by a family of DLOGTIME-uniform circuits of size $(Z(n))^{c_1}$ and depth $d_1T(n)$ for some constants c_1, d_1. One should notice that this does not say that the *class* of languages recognizable by families of logspace uniform circuits of size $Z(n)$ and depth $T(n)$ is contained in the class of languages recognizable by families of DLOGTIME-uniform circuits of size $(Z(n))^{c_2}$ and depth $d_2T(n)$ for some constants c_2, d_2.

Also, it should be mentioned that if we use Theorem 1, it is straightforward to show that DLT-U$(T(n), Z(n)) \subsetneq$ DLT-U$(T(n)\log n, (Z(n))^c)$ using the circuit simulation of PRAMs [23]. Removing this $\log n$ gap needs direct diagonalization or efficient simulation of TMs by DLOGTIME-uniform circuits, which includes several subtle details as described in the rest of the paper.

Recall that what we have to do is to construct a family of DLOGTIME-uniform circuits α_n of size $Z_2(n)$ and depth $T_2(n)$ such that the language L recognized by α_n is not recognizable by any family of DLOGTIME-uniform circuits of size $Z_1(n)$ and depth $T_1(n)$.

The overview of the circuit α_n is as follows. The circuit α_n is composed of three subcircuits, $\alpha_n^{\text{tm}}, \alpha_n^{\text{code}}$ and α_n^{sim}. The output of α_n^{tm} is 1 if and only if there exists a TM, say, T_b, such that the input string b is an encoding of T_b. Let L_{EC} be the extended connection language accepted by T_b, and let $\widehat{\beta}_n$ be the extended encoding of circuits β_n defined by L_{EC}. Circuit α_n^{code} generates the extended encoding $\widehat{\beta}_n$ of β_n by simulating T_b. (β_n may have more than $Z_1(n)$ gates, so we consider the first $Z_1(n)$ gates. If the input string b is ill-formed or if T_b does not halt within $O(\log n)$ time, then the outputs of α_n^{code} are all 0.) Strictly speaking, α_n^{code} does not generate the extended encoding $\widehat{\beta}_n$; instead, α_n^{code} checks whether each string of the form $\langle n, g, p, y\rangle$ is in L_{EC}. Since the number of different strings of the form $\langle n, g, p, y\rangle$ is roughly $(Z(n))^3$, α_n^{code} outputs a string over $\{0, 1\}$ of length about $(Z(n))^3$, where 1 (0) represents a string $\langle n, g, p, y\rangle$ is (is not) in L_{EC}. Therefore, α_n^{code} contains about $(Z(n))^3$ copies of the TM simulators.

Circuit α_n^{sim} outputs 1 if the circuit β_n has depth at most $T_1(n)$ and outputs 1. The output of α_n is 1 if and only if the outputs of α_n^{tm} and α_n^{sim} are 1 and 0, respectively. Finally, we will show that the language accepted by α_n cannot be accepted by any family of DLOGTIME-uniform circuits of size $Z_1(n)$ and depth $T_1(n)$ (see Lemma 1).

It is known [11] that there exists a DLOGTIME-uniform universal circuit U of depth $O(T_1(n))$ and size $O((Z_1(n))^3 T_1(n)/\log Z_1(n))$ such that if the extended encoding of $\widehat{\beta}_n$ of any circuit β_n of size $Z_1(n)$ and depth $T_1(n)$ is given as input, then the circuit U simulates β_n. Although the encoding inputs to U is slightly different from α_n^{sim}, the construction of α_n^{sim} is very similar to U and therefore we omit α_n^{sim}. We shall start with circuit α_n^{tm}.

Circuits α_n^{tm}: The output of α_n^{tm} is 1 if and only if there exists a TM T_b such that the input string b is an encoding of T_b. First, we must fix the encoding rule for TMs. Suppose that a TM has states $s_1, s_2, \ldots$ and uses 0, 1 as its tape-symbols. Let k be the minimum integer such that the numbers of tapes and states are less than k. State s_i is encoded into string $1^i 00\cdots 0$ of length k. Tape symbols 0 and 1 are encoded into $100\cdots 0$ and $110\cdots 0$ of length k, respectively. Strings $100\cdots 0$ and $110\cdots 0$ of length k also represent that the head is moved to the left and right, respectively. For example, if $k = 4$, we encode the next move function $\delta(s_3, 0, 1, 0) = (s_1, 1, 1, 0, L, R, L)$ into the following string of length $3k^2 + 2k$:

$$\overbrace{\underbrace{1110}_{s_3}\underbrace{1000}_{0}\underbrace{1100}_{1}\underbrace{1000}_{0}}^{\text{length } k^2+k}\overbrace{\underbrace{1000}_{s_1}\underbrace{1100}_{1}\underbrace{1100}_{1}\underbrace{1000}_{0}\underbrace{1000}_{L}\underbrace{1100}_{R}\underbrace{1000}_{L}}^{\text{length } 2k^2+k}$$

Note that each substring of length k consists of 1's followed by at least one 0. The encoding of the TM is a concatenation of the encodings of the next move functions followed by substring 1^k which is further followed by an arbitrary long string. String 1^k indicates the terminal of sequence of the next move functions. Also, all next move functions of T_b must appear in the prefix of length $\psi(n)$. ($\psi(n)$ is any slowly growing function computable by an $O(\log n)$-time DTM. We will fix $\psi(n)$ in Lemma 1.)

Circuit α_n^{tm} checks whether there exists an integer k such that the input b is an encoding of some TM with at most k tapes and k states. α_n^{tm} has circuits $c_n^{\text{tm}}(1), c_n^{\text{tm}}(2), \ldots, c_n^{\text{tm}}(k), \ldots, c_n^{\text{tm}}(\psi(n))$, where each $c_n^{\text{tm}}(k)$ checks whether b is an encoding of some TM with at most k tapes and k states. α_n^{tm} outputs 1 iff there exists an integer k such that circuit $c_n^{\text{tm}}(k)$ outputs 1. The structure of $c_n^{\text{tm}}(k)$ is as follows. Each circuit $c_n^{\text{tm}}(k)$ decides whether the input b contains at least one substring 1^k (which indicates the terminal). Let b_1 be the maximum prefix of b which does not contain 1^k. The string b is an encoding of a DTM if the prefix b_1 satisfies the following conditions: (i) b_1 consists of substrings of length $3k^2 + 2k$, (ii) each substring consists of $3k+2$ *blocks* of length k, and (iii) each block consists of 1's followed by 0's. The values of $3k^2+2k, 3k+2$, and $\psi(n)$ can be computed by an $O(\log n)$-time DTM. Once those values are known and held in storage tapes, the above structure can be checked by a single scan just as finite automata. Thus, α_n^{tm} can be a DLOGTIME-uniform circuit.

If no such k exists, α_n^{tm} outputs 0, and thus the output of α_n becomes 0, regardless of the outputs of α_n^{code} and α_n^{sim}. Therefore, in the following, we can assume that the input b meets the conditions.

Circuits α_n^{code}: Recall that L_{EC} is the extended connection language accepted by T_b, and β_n is the circuit whose extended connection language is L_{EC}. In order to find the type of gate $g(p)$ of β_n, we must decide whether $\langle n, g, p, y\rangle$ is in L_{EC} for each $y \in \{x, \wedge, \vee, \neg\}$ by simulating T_b. Thus, α_n^{code} has the following subcircuits $c_n^{\text{type}}(k, g, p, y)$ for all k and all $y \in \{x, \wedge, \vee, \neg\}$ such that each $c_n^{\text{type}}(k, g, p, y)$

checks whether $\langle n,g,p,y\rangle$ is in L_{EC}.

For simplicity, we consider $k$-tape $k$-state TM T_b which uses 0 and 1 as its tape symbols. We represent the configuration of T_b at step $t$ by four words

$$\begin{gathered} s(k,g,p,y;t;j_1,\ldots,j_k),\\ h(k,g,p,y;t,i,j_i;j_1,\ldots,j_k),\\ w_0(k,g,p,y;t,i,j;j_1,\ldots,j_k),\\ w_1(k,g,p,y;t,i,j;j_1,\ldots,j_k), \end{gathered}$$

where $s(k,g,p,y;t;j_1,\ldots,j_k)$ is a $(\log k)$-bit binary word, and the remaining three words are single bits. (In the circuit, each bit is represented by, e.g., a single AND gate of fan-in 1.) $s(k,g,p,y;t;j_1,\ldots,j_k)$ represents the state of T_b at step $t$ if $j_1,\ldots,j_k$ coincide with the head positions. For example, $s(k,g,p,y;0;j_1,\ldots,j_k)$ represents the initial state if $j_1 = \cdots = j_k = 1$ (i.e., every head is placed at the first cell at step 0); otherwise all bits of $s(k,g,p,y;0;j_1,\ldots,j_k)$ are 0. If the $j$th cell of the $i$th tape of T_b contains symbol 0 (1) and if $j_1,\ldots,j_k$ coincide with the head positions, then $w_0(k,g,p,y;t,i,j;j_1,\ldots,j_k) = 1$ $(w_1(k,g,p,y;t,i,j;j_1,\ldots,j_k) = 1)$; otherwise $w_0(k,g,p,y;t,i,j;j_1,\ldots,j_k) = 0$ $(w_1(k,g,p,y;t,i,j;j_1,\ldots,j_k) = 0)$. If the head of the $i$th tape of T_b is placed at the j_ith cell and if $j_1,\ldots,j_k$ coincide with the head positions, then $h(k,g,p,y;t,i,j_i;j_1,\ldots,j_k) = 1$; otherwise $h(k,g,p,y;t,i,j_i;j_1,\ldots,j_k) = 0$.

Recall that the next move function is encoded by a string of length $3k^2+2k$ and that this string is composed of $3k+2$ blocks of length $k$. We transform each block of the encoding of the next move function, say, δ_f, $f = 1,2,\ldots$, by the following words

$$q_1(k,f), a(k,f;i), q_2(k,f), b(k,f;i), d(k,f;i),$$

where $q_1(k,f)$ and $q_2(k,f)$ are $(\log k)$-bit binary words, and $a(k,f;i), b(k,f;i)$ and $d(k,f;i)$ are single-bits. Those words mean that if the state is $q_1(k,f)$ and the $i$th head is reading symbol $a(k,f;i)$ for each $i$, then the state is changed into $q_2(k,f)$ and the $i$th head writes $b(k,f;i)$ and moves to the left (right) if $d(k,f;i) = 0$ $(d(k,f;i) = 1)$. This transformation is done by a DLOGTIME-uniform circuit whose depth is roughly $\log k$ (details are omitted).

Now we show how to simulate a single step of TM T_b. If $t = 0$, $s(k,g,p,y;0;1,1,\ldots,1)$ represents the initial state, $w_0(k,g,p,y;0,1,j;1,1,\ldots,1)$ and $w_1(k,g,p,y;0,1,j;1,1,\ldots,1)$ for $j \geq 1$ contain the input string $\langle n,g,p,y\rangle$ in binary, and $h(k,g,p,y;0,i,1;1,1,\ldots,1) = 1$. The remaining words

$$\begin{gathered} s(k,g,p,y;0;j_1,\ldots,j_k),\\ h(k,g,p,y;0,i,j_i;j_1,\ldots,j_k),\\ w_0(k,g,p,y;0,i,j;j_1,\ldots,j_k),\\ w_1(k,g,p,y;0,i,j;j_1,\ldots,j_k) \end{gathered}$$

are set to be 0. In the following, we show the connection between steps $t$ and $t+1$.

First of all, the following circuit determines whether $j_1,j_2,\ldots,j_k$ coincide with the head positions.

$$\begin{aligned} &heads(k,g,p,y;t;j_1,\ldots,j_k)\\ &\quad = \bigwedge_{i=1}^{k} \Big(h(k,g,p,y;t,i,j_i;j_1,\ldots,j_k)\Big) \end{aligned}$$

This is an AND gate of fan-in $k$ and is replaced by a depth-$(\log k)$ circuit of fan-in 2. The following circuit outputs 1 iff two binary $k$-bit words y,z are the same.

$$EQ(y,z) = \bigwedge_{i=1}^{k} \Big(y_i z_i \vee (\neg y_i)(\neg z_i)\Big),$$

where y_i and z_i are the $i$th bit of $y$ and $z$, respectively. We compare the current state and $q_1(k,f)$ by

$$\begin{aligned} &cmp\text{-}state(k,g,p,y,f;t;j_1,\ldots,j_k)\\ &\quad = EQ\Big(s(k,g,p,y;t;j_1,\ldots,j_k), q_1(k,f)\Big)\\ &\quad \wedge heads(k,g,p,y;t;j_1,\ldots,j_k). \end{aligned}$$

Since $s(k,g,p,y;t;j_1,\ldots,j_k)$ and $q_1(k,f)$ are $(\log k)$-bit binary words, this comparison needs depth roughly $\log\log k$. We then compare the symbol read by the $i$th head of T_b and $a(k,f;i)$ by

$$\begin{aligned} &cmp\text{-}sybl_1(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k)\\ &\quad = w_1(k,g,p,y;t,i,j_i;j_1,\ldots,j_k) \wedge a(k,f;i)\\ &\quad \wedge heads(k,g,p,y;t;j_1,\ldots,j_k),\\ &cmp\text{-}sybl_0(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k)\\ &\quad = w_0(k,g,p,y;t,i,j_i;j_1,\ldots,j_k) \wedge \neg a(k,f;i)\\ &\quad \wedge heads(k,g,p,y;t;j_1,\ldots,j_k). \end{aligned}$$

We define $cmp\text{-}sybl(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k)$ as

$$\begin{aligned} &cmp\text{-}sybl(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k)\\ &\quad = cmp\text{-}sybl_0(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k)\\ &\quad \vee cmp\text{-}sybl_1(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k). \end{aligned}$$

Then $cmp\text{-}sybl(k,g,p,y,f;t,i,j_i;j_1,\ldots,j_k) = 1$ iff the $i$th head of T_b is reading the symbol which coincides with $a(k,f;i)$. Therefore, the current configuration agrees with the next move function δ_f iff the following $agree(k,g,p,y,f;t;j_1,\dots,j_k)=1$.

$$agree(k,g,p,y,f;t;j_1,\dots,j_k)$$
$$= cmp\text{-}state(k,g,p,y,f;t;j_1,\dots,j_k)$$
$$\wedge\left(\bigwedge_{i=1}^{k} cmp\text{-}sybl(k,g,p,y,f;t,i,j_i;j_1,\dots,j_k)\right)$$

Let d_i be a single bit, and $d'_i=-1$ if $d_i=0$ and $d'_i=1$ if $d_i=1$. In the following, $(d_1,d_2,\dots,d_k)$ means that the $i$th head is moved to the left (resp. right) if $d_i=0$ (resp. $d_i=1$). We define $move(k,f;d_1,d_2,\dots,d_k)$ as

$$move(k,f;d_1,d_2,\dots,d_k)=\bigwedge_{i=1}^{k} EQ\Big(d_i,d(k,f;i)\Big).$$

We define $s(k,g,p,y,f;t+1;j_1,\dots,j_k;d_1,\dots,d_k)$ as

$$q_2(k,f)\wedge agree(k,g,p,y,f;t;j_1-d'_1,\dots,j_k-d'_k)$$
$$\wedge move(k,f;d_1,\dots,d_k).$$

Now the next state are updated by

$$s(k,g,p,y;t+1;j_1,\dots,j_k)=$$
$$\bigvee_{d_1,\dots,d_k\in\{0,1\}}\left(\bigvee_{f=1}^{k\cdot 2^k} s(k,g,p,y,f;t+1;j_1,\dots,j_k;d_1,\dots,d_k)\right),$$

where $k\cdot 2^k$ is the number of the next move functions. This is an OR gate of fan-in $2^k k 2^k$, and thus this can be replaced by a fan-in 2 circuit of depth $2k+\log k$.

Then, we define $w_1(k,g,p,y,f;t+1,i,j;j_1+d_1,\dots,j_k+d_k;d_1,\dots,d_k)$ as

$$\Big(b(k,f;i)\wedge agree(k,g,p,y,f;t;j_1,\dots,j_k)$$
$$\wedge h(k,g,p,y;t,i,j_i;j_1,\dots,j_k)\wedge move(k,f;d_1,\dots,d_k)\Big)$$
$$\vee\Big(w_1(k,g,p,y;t,i,j;j_1,\dots,j_k)$$
$$\wedge\neg h(k,g,p,y;t,i,j_i;j_1,\dots,j_k)\wedge move(k,f;d_1,\dots,d_k)\Big).$$

Tape symbols are updated by

$$w_1(k,g,p,y;t+1,i,j;j_1,\dots,j_k)=$$
$$\bigvee_{d_1,\dots,d_k\in\{0,1\}}\left(\bigvee_{f=1}^{k\cdot 2^k} w_1(k,g,p,y,f;t+1,i,j;j_1,\dots,j_k;d_1,\dots,d_k)\right).$$

$w_0(k,g,p,y,f;t+1,i,j;j_1,\dots,j_k;d_1,\dots,d_k)$ and $w_0(k,g,p,y;t+1,i,j;j_1,\dots,j_k)$ are defined similarly.

The head positions are updated if the head positions at step $t+1$ are adjacent to the positions at step $t$. We define $h(k,g,p,y,f;t+1,i,j_i;j_1,j_2,\dots,j_k;d_1,d_2,\dots,d_k)$ as

$$h(k,g,p,y;t,i,j_i-d'_i;j_1-d'_1,\dots,j_k-d'_k)$$
$$\wedge agree(k,g,p,y,f;t;j_1-d'_1,\dots,j_k-d'_k)$$
$$\wedge move(k,f;d_1,\dots,d_k).$$

$h(k,g,p,y,f;t+1,i,j_i;j_1,j_2,\dots,j_k;d_1,d_2,\dots,d_k)=1$ iff the $i$th head is placed at $(j_i-d'_i)$th cell at step $t$ and is moved to j_ith cell at step $t+1$ for each $i$. Then, the head positions are updated by

$$h(k,g,p,y;t+1,i,j_i;j_1,j_2,\dots,j_k)=$$
$$\bigvee_{d_1,\dots,d_k\in\{0,1\}}\left(\bigvee_{f=1}^{k\cdot 2^k} h(k,g,p,y,f;t+1,i,j_i;j_1,\dots,j_k;d_1,\dots,d_k)\right).$$

As shown in Lemma 1, it is enough that $\alpha_n^{\rm code}$ simulates $\psi(n)\log n$ steps of T_b. A single step of the simulation needs depth $c_1 k$ for some constant c_1. Therefore, $\alpha_n^{\rm code}$ has depth $c_1 k\psi(n)\log n$ in total.

Each gate has its own name, and each name is represented by a binary string of length $c_2\log n$ for some constant c_2. The depth of the connection between step $t$ and step $t+1$ is a polynomial in $k$. Therefore, the type and gate number of the gate reached by following some path p, $|p|\le\log Z(n)$, from each gate of $\alpha_n^{\rm code}$ can be computed by an $O(\log n)$-time DTM. Hence, α_n is a DLOGTIME-uniform circuit. The following lemma concludes the proof.

Lemma 1. *Any family of DLOGTIME-uniform circuits of size $Z_1(n)$ and depth $T_1(n)$ cannot recognize the language which is recognized by the above-defined α_n of size $Z_2(n)$ and depth $T_2(n)$.*

Proof. Assume for contradiction that there exists a family of DLOGTIME-uniform circuits, say, β_n, of size $Z_1(n)$ and depth $T_1(n)$ such that β_n can accept the language accepted by α_n. Since β_n is DLOGTIME-uniform, there exists an $O(\log n)$-time DTM T_b which recognizes the extended connection language $L_{\rm EC}$ of β_n. Consider a sufficiently long string $b$ such that the encoding of T_b appears in the prefix of length $\psi(n)$. If we define $\alpha_n^{\rm code}$ by using an appropriate slowly growing function $\psi(n)$ computable by an $O(\log n)$-time DTM (e.g., $\psi(n)=\min(\log^* n,\sqrt{T_1(n)/\log n})$), then the depth $c_1 k\psi(n)\log n$ of $\alpha_n^{\rm code}$ becomes at most $dT_1(n)$. (Note that $k$ is at most $\psi(n)$ and $T_1(n)$ is not bounded by $O(\log n)$.)

If such a long string $b$ is given to α_n as its input, then (i) the output of $\alpha_n^{\rm tm}$ is 1, (ii) $\alpha_n^{\rm code}$ correctly outputs the extended encoding $\widehat{\beta}_n$ of β_n, and therefore (iii) $\alpha_n^{\rm sim}$ outputs 1 if and only if β_n outputs 1. Recall that the output of α_n is 1 if and only if the outputs of $\alpha_n^{\rm tm}$ and $\alpha_n^{\rm sim}$ are 1 and 0. Therefore, α_n outputs 1 if and only if β_n outputs 0, a contradiction. □

Acknowledgments

We would like to thank Eric Allender for valuable suggestions and comments on the initial drafts of this paper. This work originated from the discussion with him while the first author visited Rutgers University. Also, thanks are due to an anonymous referee who gave us the suggestion on the universal circuits of Cook and Hoover. This research was supported in part by Scientific Research Grant, Ministry of Education, Japan.

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley, Reading, MA, 1974.

[2] M. Ajtai, "Σ_1^1-formulae on finite structure," *Ann. Pure Appl. Logic*, Vol. 24, pp. 1-48, 1983.

[3] D.A. Barrington, "Bounded-width polynomial-size branching programs can recognize exactly those languages in NC^1," *J. Comput. System Sci.*, Vol. 38, pp. 150-164, 1989.

[4] D.A.M. Barrington, K. Compton, H. Straubing, and D. Thérien, "Regular Languages in NC^1," *J. Comput. System Sci.*, Vol. 44, pp. 478-499, 1992.

[5] D.A.M. Barrington, N. Immerman, and H. Straubing, "On uniformity within NC^1," *J. Comput. System Sci.*, Vol. 41, pp. 274-306, 1990.

[6] D.A.M. Barrington, H. Straubing, and D. Thérien, "Non-uniform Automata over groups," *Inform. and Comput.*, Vol. 89, pp. 109-132, 1990.

[7] D.A.M. Barrington and D. Thérien, "Finite monoids and the fine structure of NC^1," *J. Assoc. Comput. Mach.*, Vol. 35, No. 4, pp. 941-952, 1988.

[8] A. Borodin, "On relating time and space to size and depth," *SIAM J. Comput.*, Vol. 6, No. 4, pp. 733-743, 1977.

[9] S.R. Buss, S.A. Cook, A. Gupta, and V. Ramachandran, "An optimal parallel algorithm for formula evaluation," *SIAM J. Comput.*, Vol. 14, pp. 755-780, 1992.

[10] S.A. Cook, "A taxonomy of problems with fast parallel algorithms," *Inform. and Control*, Vol. 64, pp. 2-22, 1985.

[11] S.A Cook and H.J. Hoover, "A depth-universal circuit," *SIAM J. Comput.*, Vol. 14, No. 4, pp. 833-839, 1985.

[12] S.A. Cook and R.A. Reckhow, "Time bounded random access machines," *J. Comput. System Sci.*, Vol. 7, pp. 354-375, 1973.

[13] M. Furst, J. Saxe and M. Sipser, "Parity, circuits, and the polynomial time hierarchy," *Mach. Systems Theory*, Vol. 17, pp. 12-27, 1984.

[14] K. Iwama and C. Iwamoto, "Extended graph connectivity and its gradually increasing parallel complexity," in: *Proc. Fifth Annual International Symposium on Algorithms and Computation* (*Lecture Notes in Computer Science* 834), pp. 478-486, 1994.

[15] K. Iwama and C. Iwamoto, "α-connectivity: A gradually non-parallel graph problem," to appear in the *Journal of Algorithms*, Vol. 20, No. 3, 1996.

[16] K. Iwama, C. Iwamoto, and M. Morshed, "Time lower bounds do not exist for CRCW PRAMs," to appear in *Theoretical Computer Science*, Vol. 160, 1996.

[17] D.S. Johnson, "A catalog of complexity classes", in: J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. A* (North-Holland, Amsterdam, 1990) pp. 69-161.

[18] S. Khuller and B. Schieber, "Efficient parallel algorithms for testing k-connectivity and finding disjoint s-t paths in graphs," *SIAM J. Comput.*, Vol. 20, No. 2, pp. 352-375, 1991.

[19] W.W. Kirchherr, "A hierarchy theorem for PRAM-based complexity classes," in: *Proc. 8th Conference on Foundations of Software Technology and Theoretical Computer Science* (*Lecture Notes in Computer Science* 338), pp. 240-249, 1988.

[20] N. Pippenger and M.J. Fischer, "Relations among complexity measures," *J. Assoc. Comput. Mach.*, Vol. 26, No. 2, pp. 361-381, 1979.

[21] W.L. Ruzzo, "On uniform circuit complexity," *J. Comput. System Sci.*, Vol. 22, pp. 365-383, 1981.

[22] M. Sipser, "Borel sets and circuit complexity," in: *Proc. 15th Ann. ACM Symp. on Theory of Computing*, pp. 61-69, 1983.

[23] L. Stockmeyer and U. Vishkin, "Simulation of parallel random access machines by circuits," *SIAM J. Comput.*, Vol. 13, No. 2, pp. 409-422, 1984.