WS1S is non-elementary: A Summary of Meyer's Proof

Ting Zhang

June 6, 2001

1 Preliminaries

In this section we briefly review some standard definitions and notations in formal language and recursion theories.

1.1 Formal Language Theory

Let Σ be a finite alphabet. A word is a finite sequence of symbols over Σ. Empty word is denoted by ϵ. Σ^* is the set of all words over Σ. We say that A is a (formal) language if $A \subseteq \Sigma^*$. Let $A, B \subseteq \Sigma$. $A \cdot B$ is the concatenation operation. $A \cup B$ (also written $A + B$), $A \cap B$ and $\neg A$ are set-theoretic operations. Given $A \subseteq \Sigma^*$, the following are standard definitions in the literature.

$$A^0 = \{\epsilon\},\ A^{n+1} = A^n \cdot A,\ A^* = \cup_{n=0}^{\infty} A^n,\ A^+ = \cup_{n=1}^{\infty} A^n.$$

We introduce $\natural$-operation defined by the rule:

$$A^\natural = \{\sigma \mid \sigma \in \Sigma^* \text{ and } \exists \sigma' \in A \text{ s.t. } |\sigma| = |\sigma'|\}.$$

If $\mathcal{A}_A$ is an automaton recognizing $A \subseteq \Sigma^*$, we can construct an automaton $\mathcal{A}_{A^\natural}$ where $\mathcal{A}_{A^\natural}$ is the same as $\mathcal{A}_A$ except that every transition edge is universal for all input symbols. Apparently $\mathcal{A}_{A^\natural}$ recognizes $A^\natural$. Hence all preceding operations preserve language regularity. We define γ-expressions which are constructed inductively from these operations.

Definition 1.1 (γ-expression). *A γ-expression over Σ is defined inductively as follows:*

(1) $\mathbf{a}$ *is a γ-expression for* $\mathbf{a} \in \Sigma$ *and* $L(\mathbf{a}) = \{\mathbf{a}\}$.

(2) If α, β are γ-expressions, then $\alpha \cdot \beta$, $\alpha \cap \beta$, $\alpha \cup \beta$, $\neg\alpha$ and $\alpha^\natural$ are all γ-expressions, and

$$\begin{aligned} L(\alpha \cdot \beta) &= L(\alpha) \cdot L(\beta) \\ L(\alpha \cup \beta) &= L(\alpha) \cup L(\beta) \\ L(\alpha \cap \beta) &= L(\alpha) \cap L(\beta) \\ L(\neg\alpha) &= \Sigma^* - L(\alpha) \\ L(\alpha^\natural) &= L(\alpha)^\natural \end{aligned}$$

By $\mathbf{SAT}(\Sigma)$ we denote the set of γ-expressions whose language is non-empty.

1.2 Recursion Theory

Definition 1.2 (Elementary Recursive [Odi99]). *A class $\mathcal{E}$ of elementary recursive functions is the smallest class of functions:*

(1) containing $x+1$, $x-y$ and $x+y$,

(2) closed under composition,

(3) closed under bounded sum and product, defined by

$$\sum_{i=0}^{y} \varphi(\vec{x}, i) \text{ and } \prod_{i=0}^{y} \varphi(\vec{x}, i).$$

A language is *elementary* if its characteristic function is. Let **Th**(**WS1S**) be the set of all true **WS1S** sentences. We shall show that **Th**(**WS1S**) is not elementary. Our computation model is the standard **one-tape**[1]**deterministic Turing machine**. Let $\varphi(n) : \mathbb{N} \mapsto \mathbb{N}$ be a computable function. We say that a Turing machine $\mathcal{M}$ is $\varphi(n)$-*time bounded* (resptively, $\varphi(n)$-*space bounded*) if $\mathcal{M}$ halts after at most $\varphi(n)$ moves (resptively, after visiting at most $\varphi(n)$ tape squares) for any input of length $n \geq 0$. We denote by $\mathbf{SPACE}(\varphi(n))$ (respectively, $\mathbf{TIME}(\varphi(n))$) the class of all recursive functions computable by $\varphi(n)$-*time bounded* Turing machines (respectively, $\varphi(n)$-*space bounded* Turing machines.) Let us define a series of functions as follows:

$$\begin{aligned} E_0(n) &= n \\ E_{k+1}(n) &= 2^{E_k(n)} \end{aligned}$$

A well-known characteristic of $\mathcal{E}$ was due to R. W. Ritchie.

Theorem 1.1 (Elementary Function Characterization [Rit63]). *A language is in $\mathcal{E}$ iff it is in $\mathbf{SPACE}(E_k(n))$ for some fixed $k > 0$ and all inputs of length $n \geq 0$.*

In [Rit63], Ritchie considered an infinite hierarchy of strictly inclusive classes of functions $\{F_i\}_{i \geq 0}$. For each $i > 0$, F_i is defined to be a class of functions whose space complexity functions are contained in F_{i-1}. Theorem 1.1 follows from the fact that for each $i > 1$, the space complexity of functions in F_i is bounded by E_{i-1} and the union of the hierarchy $\mathcal{F} = \cup_{i=0}^{\infty} F_i = \mathcal{E}$.

Definition 1.3 (Polynomial-Time Reduction). *Let $A_1 \subseteq \Sigma_1^*, A_2 \subseteq \Sigma_2^*$ be two languages. We say that A_1 is polynomial-time reducible to A_2 (written $A_1 \prec_p A_2$) iff there is a polynomial-time bounded Turing machine $\mathcal{M}$ which outputs word $y \in \Sigma_2^*$ with any input $x \in \Sigma_1^*$ such that $x \in A_1$ iff $y \in A_2$.*

Clearly, for any $k > 0$, $A_1 \prec_p A_2$ and $A_2 \in \mathbf{SPACE}(E_k(n))$ imply $A_1 \in \mathbf{SPACE}(E_k(n))$.

Definition 1.4 (Space Constructible [SHI65]). *A function $f : \mathbb{N} \mapsto \mathbb{N}$ is space constructible iff there is a Turing machine $\mathcal{M}$ which halts after using exact $\varphi(n)$ tape squares for any input of length $n \geq 0$.*

Space constructibility rules out the pathological functions (see **Gap Theorem**) to obtain straightly inclusive complexity hierarchies.

[1] Here we don't need to consider space complexity smaller than input size.

Theorem 1.2 (Space Hierarchy Theorem). *If $\varphi(n)$ is a space constructible function, then there exists a language A which is $\varphi(n)$-space computable and is not $\psi(n)$-space computable for all functions $\psi(n)$ such that*

$$\lim_{n\to\infty} \frac{\psi(n)}{\varphi(n)} = 0$$

The result was proved by the diagonalization argument. It can be shown that there exists a $\varphi(n)$-space bounded Turing machine $\mathcal{M}$ which simulates all $\psi(n)$-space bounded Turing machines and differs from each of them. It follows that $L(\mathcal{M}) \in \mathbf{SPACE}(\varphi(n)) - \mathbf{SPACE}(\psi(n))$.

2 Proof

2.1 Proof Outline

The main idea of the proof is to show that **WS1S** has very strong encoding ability, that is, **WS1S** can efficiently encode all computations of elementary space-bounded Turing machines. Therefore, **Th**(**WS1S**) itself must be very difficult to decide.

Lemma 2.1. *If for every $k > 0$ and for any language $A \subseteq \{\mathbf{0},\mathbf{1}\}^*$ such that $A \in \mathbf{SPACE}(E_k(n))$, we have $A \prec_p \mathbf{Th}(\mathbf{WS1S})$, then $\mathbf{Th}(\mathbf{WS1S})$ is non-elementary.*

Proof Sketch. It is well-known that constant functions, $x+y$, $x \cdot y$, x^y are all space-constructible and composition preserve space-constructibility. Hence $E_{k+1}(n)$ is space-constructible for any fixed k, and

$$\lim_{n\to\infty} \frac{E_k(n)}{E_{k+1}(n)} = 0.$$

By Theorem 1.2, there exists a language $A \subseteq \{\mathbf{0},\mathbf{1}\}^*$ such that $A \in \mathbf{SPACE}(E_{k+1}(n))$, but $A \notin \mathbf{SPACE}(E_k(n))$. As $A \prec_p \mathbf{Th}(\mathbf{WS1S})$, it must be true that $\mathbf{Th}(\mathbf{WS1S}) \notin \mathbf{SPACE}(E_k(n))$ for any fixed $k > 0$. By Theorem 1.1, we conclude that $\mathbf{Th}(\mathbf{WS1S})$ is not elementary recursive. □

In the following subsection, we establish in two steps the desired reduction in the lemma premise.

2.2 Reductions

Let A be the language aforementioned in Lemma 2.1. We shall show that $A \prec_p \mathbf{SAT}(\Sigma)$ and $\mathbf{SAT}(\Sigma) \prec_p \mathbf{Th}(\mathbf{WS1S})$. By the transitivity of $\prec_p$, we will have $A \prec_p \mathbf{Th}(\mathbf{WS1S})$.

Lemma 2.2. $\mathbf{SAT}(\Sigma) \prec_p \mathbf{Th}(\mathbf{WS1S})$

Proof Sketch. For simplicity, we exclude ϵ from our discussions. Also, we assume that $\Sigma = \{\mathbf{0},\mathbf{1}\}$. Using suitable encoding, there is no conceptual difficulty to generalize the result to an arbitrary Σ. We construct a reduction function which translates each γ-expression α to a **WS1S** formula $F_\alpha(i,j,K)$ where i,j are first-order variable and K is a second-order variable. In standard interpretation, i,j are natural numbers and K is a finite subset of natural numbers. K can be viewed as an infinite word σ over Σ^* with only finite occurrences of $\mathbf{1}$'s such that the i^{th} symbol σ_i is 1 iff $i \in K$. Intuitively $F_\alpha(i,j,K)$ says that $i < j$ and the subword $\sigma_i\sigma_{i+1}\cdots\sigma_{j-1} \in L(\alpha)$. $F_\alpha(i,j,K)$ is constructed inductively on structure of γ-expressions.

(1) If $\alpha = \mathbf{0}$ or $\mathbf{1}$, then

$$F_{\mathbf{0}}(i,j,K) \stackrel{\text{def}}{=} (j = i+1) \wedge (i \notin K),$$
$$F_{\mathbf{1}}(i,j,K) \stackrel{\text{def}}{=} (j = i+1) \wedge (i \in K),$$

(2) If $\alpha = \beta \cdot \delta$, then

$$F_\alpha(i,j,K) \stackrel{\text{def}}{=} \exists k(i \le k \le j \wedge F_\beta(i,k,K) \wedge F_\delta(k,j,K)),$$

(3) If $\alpha = \beta^\natural$, then

$$F_\alpha(i,j,K) \stackrel{\text{def}}{=} \exists K'(F_\beta(i,j,K')),$$

(4) If $\alpha = \beta \cup \delta$, $\beta \cap \delta$ or $\neg\beta$, then respectively,

$$F_\alpha(i,j,K) \stackrel{\text{def}}{=} F_\beta(i,j,K) \vee F_\delta(i,j,K),$$
$$F_\alpha(i,j,K) \stackrel{\text{def}}{=} F_\beta(i,j,K) \wedge F_\delta(i,j,K),$$
$$F_\alpha(i,j,K) \stackrel{\text{def}}{=} \neg F_\beta(i,j,K).$$

By induction it is easily shown that

$$L(\alpha) \neq \emptyset \iff \exists i \exists j \exists K[F_\alpha(i,j,K)].$$

□

Next we show that γ-expression has very strong encoding ability such that it can encode any computations of elementary space-bounded Turing machines and the encoding can be done in polynomial time w.r.t. the input length. Let $\mathcal{M}$ be any Turing machine with an input alphabet $\Sigma_I = \{\mathbf{0}, \mathbf{1}\}$, a tape alphabet $\Sigma = \{\mathbf{0}, \mathbf{1}, \flat, \sharp\}$ and a state set Q. An *instantaneous description* (ID) of $\mathcal{M}$ is a word σ in $(\Sigma \cup Q)^*$ which contains exactly one symbol in Q. Here we require that IDs have the uniform length. $Next_\mathcal{M}(\sigma)$ is defined to be the immediate successor ID of σ. (We assumed that $\mathcal{M}$ is deterministic.) $Next_\mathcal{M}(\sigma)$ is undefined if σ contains a halting state or its head is going to across the ID boundary (I.e., trying to move right (left) at leftmost (rightmost) of σ.) Let $Next_\mathcal{M}(\sigma, 0) = \sigma$ if σ is an ID and undefined otherwise. Let $Next_\mathcal{M}(\sigma, n+1) = Next_\mathcal{M}(Next_\mathcal{M}(\sigma, n))$. A computation $Comp_\mathcal{M}(\sigma)$ is a partial function on ID σ defined as follows:

$$Comp_\mathcal{M}(\sigma) = \{\sharp \cdot Next_\mathcal{M}(\sigma,0) \cdot \sharp \cdot Next_\mathcal{M}(\sigma,1) \cdot \sharp \cdots \sharp \cdot Next_\mathcal{M}(\sigma,n) \cdot \sharp\}.$$

where $Next_\mathcal{M}(\sigma, n)$ contains a halting state q_f. As $\mathcal{M}$ is deterministic, $Comp_\mathcal{M}(\sigma)$ is either singleton or $\emptyset$. We may use $Comp_\mathcal{M}(\sigma)$ to denote the word in it whenever $Comp_\mathcal{M}(\sigma) \neq \emptyset$. The meaning should be clear from the context. Assume that $\sigma' = Next_\mathcal{M}(\sigma)$ is defined. it is well-known that any four consecutive symbols σ_i, σ_{i+1}, σ_{i+2}, σ_{i+3} uniquely determine the symbol σ'_{i+1}. Let $\delta_\mathcal{M} : \Sigma^4 \mapsto \Sigma$ be such a partial function determined by $\mathcal{M}$. $\delta_\mathcal{M}(\eta_0 \cdot \eta_1 \cdot \eta_2 \cdot \eta_3)$ is undefined if $\eta_0 \cdot \eta_1 \cdot \eta_2 \cdot \eta_3$ contains a halting state or a state without any transitions. Assume that $Comp_\mathcal{M}(\sigma)$ is defined. It is easily seen that if $\eta_0, \eta_1, \eta_2, \eta_3$ are the $i^{th}, (i+1)^{th}, (i+2)^{th}, (i+3)^{th}$ symbols of $Comp_\mathcal{M}(\sigma)$, then $\delta_\mathcal{M}(\eta_0 \cdot \eta_1 \cdot \eta_2 \cdot \eta_3)$[2]are the $(|\sigma| + i + 2)^{th}$ symbol of $Comp_\mathcal{M}(\sigma)$ providing $i \le |Comp_\mathcal{M}(\sigma)| - |\sigma| - 2$.

[2]Actually, we need to extend $\delta_\mathcal{M}$ to deal with the situation that $\eta_0 \cdot \eta_1 \cdot \eta_2 \cdot \eta_3$ traverses ID boundries. However, the details are of no importance.

Lemma 2.3 (Simulation Lemma). *Let $\mathcal{M}$ be a Turing machine in the preceding definition. Let α be a γ-expression over $\Gamma = (\Sigma \cup Q)$ and $L(\alpha) = \Gamma^n$ for some $n > 0$. There exists a γ-expression β over Γ such that*

$$L(\beta) = Comp_{\mathcal{M}}(\flat^n \cdot x \cdot \flat^n),$$

and β is constructible in polynomial time w.r.t. the length of $(x \cdot \sharp \cdot \alpha)$.

Proof Sketch. We shall construct a γ-expression β such that $L(\beta) = Comp_{\mathcal{M}}(\flat^n \cdot x \cdot \flat^n)$. Note that $\neg Comp_{\mathcal{M}}(\flat^n \cdot x \cdot \flat^n)$ can be characterized as follows:

(1) words that do not begin with $\sharp \cdot \flat^n \cdot x \cdot \flat^n \cdot \sharp$:

$$\begin{aligned} R_1 &= \neg(\sharp \cdot (\Gamma^n \cap \flat^*) \cdot x \cdot (\Gamma^n \cap \flat^*) \cdot \sharp \cdot \Gamma^*) \\ &= \neg(\sharp \cdot (\alpha \cap \flat^*) \cdot x \cdot (\alpha \cap \flat^*) \cdot \sharp \cdot \Gamma^*). \end{aligned}$$

(2) words that do not contain final state q_f:

$$R_2 = \neg(\Gamma^* \cdot q_f \cdot \Gamma^*).$$

(3) words that do not end with $\sharp$:

$$R_3 = \neg(\Gamma^* \cdot \sharp).$$

(4) words that do not satisfy the consecution property:

$$\begin{aligned} R_4 &= \bigcup_{\sigma_0,\sigma_1,\sigma_2,\sigma_3 \in \Gamma} [\Gamma^* \cdot \sigma_0 \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdot \Gamma^{2n+|x|-2} \cdot (\Gamma - \delta_{\mathcal{M}}(\sigma_0 \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3)) \cdot \Gamma^*] \\ &= \bigcup_{\sigma_0,\sigma_1,\sigma_2,\sigma_3 \in \Gamma} [\Gamma^* \cdot \sigma_0 \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3 \cdot \alpha \cdot \Gamma^{|x|-2} \cdot \alpha \cdot (\Gamma - \delta_{\mathcal{M}}(\sigma_0 \cdot \sigma_1 \cdot \sigma_2 \cdot \sigma_3)) \cdot \Gamma^*]. \end{aligned}$$

Obviously

$$Comp_{\mathcal{M}}(\flat^n \cdot x \cdot \flat^n) = \neg(R_1 \cup R_2 \cup R_3 \cup R_4).$$

Here operation "$*$" and "$-$" are only used for the sake of brevity. They are efficiently expressible via the basic γ-operations. Moreover all constructions only take polynomial time w.r.t. to the length of $(x \cdot \sharp \cdot \alpha)$. $\square$

Next we show that $\natural$-operation can help encode sets of words with very long uniform length.

Lemma 2.4. *For any $k \geq 0$, there exists a γ-expression α such that $L(\alpha) = \Sigma^{f_k(n)}$ where $f_k(n) \geq E_k(n)$ and α can be constructed in polynomial time w.r.t. n.*

Proof Sketch. We prove it by induction. The base case is trivial. Now assume the hypothesis holds for k. Since $E_k(n)$ is space constructible, we can construct a new Turing machine $\mathcal{M}$ which first mark down (by "**1**") exactly $E_k(n)$ tape squares and then circles within such space $E_{k+1}(n)$ steps. ($\mathcal{M}$ can do this easily by using $E_k(n)$ tape squares as a counter.) By Lemma 2.3, we can have a γ-expression β such that $L(\beta) = Comp_{\mathcal{M}}(x)$ where $x = \flat^{f_k(n)} \cdot q_0 \cdot \flat^{f_k(n)}$. $Comp_{\mathcal{M}}(x)$ is defined as $\mathcal{M}$ halts within $E_k(n) \leq f_k(n)$ tape squares. Moreover, $|Comp_{\mathcal{M}}(x)| \geq E_{k+1}(n)$ as $\mathcal{M}$ runs at least $E_{k+1}(n)$ steps. Hence $\alpha = \beta^{\natural}$. Again, all constructions are polynomial-time bounded w.r.t. n. $\square$

Lemma 2.5. *For any $A \subseteq \{\mathbf{0}, \mathbf{1}\}^*$, if $A \in \mathbf{SPACE}(E_k(n))$ for some $k \geq 0$, then there exists a finite Γ such that $A \prec_p \mathbf{SAT}(\Gamma)$.*

Proof Sketch. Let $\mathcal{M}$ be a Turing machine which recognizes A within space bound $E_k(|x|)$ for all $x \in A$. Let Γ as previously defined. By Lemma 2.4, we have a γ-expression such that $L(\alpha) = \Gamma^{f_k(|x|)}$ where $f_k(|x|) \geq E_k(|x|)$. Applying Lemma 2.3, we obtain a γ-expression β such that $L(\beta) = Comp_m(\flat^{f_k(|x|)} \cdot q_0 \cdot x \cdot \flat^{f_k(|x|)})$ where q_0 is the initial state of $\mathcal{M}$. Since $\mathcal{M}$ is space-bounded by $E_k(n)$, $Comp_m(\flat^{f_k(|x|)} \cdot q_0 \cdot x \cdot \flat^{f_k(|x|)})$ is non-empty iff x is accepted by $\mathcal{M}$. Hence, $x \in A$ iff $Comp_m(\flat^{f_k(|x|)} \cdot q_0 \cdot x \cdot \flat^{f_k(|x|)}) \neq \emptyset$ iff $L(\beta) \neq \emptyset$ iff $\beta \in \mathbf{SAT}(\Gamma)$. Once again, it is clear that the reduction can be constructed in polynomial time w.r.t. the input length. □

Theorem 2.1 ([Mey75]). Th(WS1S) *is non-elementary.*

Proof. By Lemma 2.1, 2.2 and 2.5. □

References

[Mey75] A.R. Meyer. Weak monadic second order theory of successor is not elementary recursive. In *Proceeding of Logic Colloquium*, vol. 453 of *Lecture Notes in Mathematics*, pages 132–154. Springer-Verlag, 1975.

[Odi99] P. Odifreddi. *Classical Recursion Theory II*, pages 264–286. Elsevier, 1999.

[Rit63] R.W. Ritchie. Classes of predictably computable functions. *Transactions of American Mathematical Society*, 106(1):139–173, January 1963.

[SHI65] R.E. Stearns, J. Hartmanis, and P.M.L. II. Hierarchies of memory limited computations. In *Proceeding of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 179–190. IEEE, 1965.