On Unambigous Nondeterministic Finite Automata and the Strict Tree Property

Henry N. Adorna [*]
Department of Computer Science
(Algorithms and Complexity)
Velasquez Ave., UPDiliman
Quezon City 1101
hnadorna@up.edu.ph

Nestine Hope S. Hernandez [†]
Department of Computer Science
(Algorithms and Complexity)
Velasquez Ave., UPDiliman
Quezon City 1101
nshernandez@up.edu.ph

Rex David D. Lorenzo
Department of Computer Science
(Algorithms and Complexity)
Velasquez Ave., UPDiliman
Quezon City 1101
rdlorenzo1@up.edu.ph

ABSTRACT

In this paper, we intend to discuss the class of automata between deterministic finite automata (**DFA**) and unambiguous nondeterministic finite automata (**UFA**), which has the strict tree property. We call this class the Strict Unambiguous Finite Automata (**StUFA**)[1]. A reasonable characterization of these automata based on the transition relation Δ will be provided.

Keywords

Automata, ambiguity, nondeterminism, computation tree

[*]H. Adorna is partially supported by a FRIA grant of the College of Engineering, UPDiliman.

[†]N.H. Hernandez is supported by NGSE Scholarship program of the College of Engineering, UPDiliman.

[1]This class of nondeterministic finite automata was previously called **HrTFA** in [1] and **SUFA** in [2].

1. INTRODUCTION

In [4] the following hypothesis on the computation tree of any unambiguous nondeterministic finite automata was given:

> **Strict Tree Property**
> *The computation tree of any minimal $\mathcal{M} \in$ **UFA** on an input w has exactly one path P from root to a leaf with several nondeterministic guesses and all paths having only one vertex in common with P do not contain any nondeterministic branching.*

However, [1] provided a counter example by showing a minimal unambiguous finite automaton that accepts a word wherein the computation tree disobeys the Strict Tree Property.

This (strict tree) property on the computation tree of an unambiguous nondeterministic finite automata is found to be satisfied by some automata which are different from the class of deterministic finite automata (**DFA**). This is the class of automata that we are interested in.

The paper is organized as follows: In Section 2, we provide the definitions essential to the paper, however, some terminologies which are standard in formal language theory may be found in [3]. In Section 3 we discuss the computation tree of an $\mathcal{M} \in$ **NFA.** We also introduce the basic tool we need for the main results. A counter example for the claim in [4] is provided in Section 4. We give some state complexity results in Section 5. Finally we prove our main result in Section 6.

2. DEFINITIONS

Let Σ be any finite set of symbols. We denote by Σ^* the set of all possible strings or words that can be formed from the elements of Σ. Let $A \subseteq \Sigma^*$. We define a simple model of computation that accepts those and only those strings that are in A.

DEFINITION 1. *A 5-tuple $\mathcal{M} = (Q, \Sigma, I, \Delta, F)$ is called a* **nondeterministic finite automaton** *where Q is a finite*

set of states, Σ is a finite set of alphabet, $I \subseteq Q$ is called the set of initial states, $F \subseteq Q$ is a set of accepting states and a relation $\Delta \subseteq Q \times \Sigma \times 2^Q$.

We will denote the **class of nondeterministic finite automata** *as* **NFA.**

The relation Δ is defined only on the elements of $Q \times \Sigma$, however the automaton needs to accept set of strings over Σ^*. In this case, we extend our relation to the elements of Σ^* as follows:

DEFINITION 2. *We define the relation*

$$\widehat{\Delta} \subseteq Q \times \Sigma^* \times 2^Q,$$

to be the smallest set s.t. for all $u, v \in \Sigma^*, q, q', q'' \in Q$, it satisfies the following:

(i) $\Delta \subseteq \widehat{\Delta}$,

(ii) $(q, \lambda, q) \in \widehat{\Delta}$,

(iii) $(q, u, q'), (q', v, q'') \in \widehat{\Delta} \Longrightarrow (q, uv, q'') \in \widehat{\Delta}$.

Let $w \in \Sigma^*$. The word w is accepted by an automaton $\mathcal{M}$ iff there are states $q_i \in I$ and $q_f \in F$ s.t. $(q_i, w, q_f) \cap \widehat{\Delta} \neq \emptyset$.

DEFINITION 3. *We define the* **language accepted** *by $\mathcal{M}$ as:*

$$L(\mathcal{M}) = \{\, w \in \Sigma^* \mid (I \times \{w\} \times F) \cap \widehat{\Delta} \neq \emptyset \,\}.$$

In fact, we can represent these set of strings as paths in the transition diagram of $\mathcal{M}$. These paths are concatenations of elements of $Q \cup \Sigma$. In particular, these paths are subset of $Q \cdot (\Sigma \cdot Q)^*$.

We define the following functions:

DEFINITION 4.

$$\mathrm{Proj}_Q : Q \cdot (\Sigma \cup Q)^* \longrightarrow Q^*$$

s.t. for all $x \in \Sigma$, $\mathrm{Proj}_Q(x) = \lambda$ *while for all $q \in Q$,* $\mathrm{Proj}_Q(q) = q$,

$$\mathrm{Proj}_\Sigma : Q \cdot (\Sigma \cup Q)^* \longrightarrow \Sigma^*$$

s.t. for all $q \in Q$, $\mathrm{Proj}_\Sigma(q) = \lambda$ *while for all $x \in \Sigma$,* $\mathrm{Proj}_\Sigma(x) = x$,

We denote the set of all possible paths, whose vertices are labeled with elements of Q and edges by the symbols in any $w \in \Sigma^*$ in chronological order by $\mathrm{Path}(\mathcal{M})$. More precisely, $\mathrm{Path}(\mathcal{M}) \subseteq Q \cdot (\Sigma \cdot Q)^*$. Formally,

DEFINITION 5.

$$\mathit{Path}(\mathcal{M}) = \{\, x \in Q \cdot (\Sigma \cdot Q)^* \mid \exists\, w \in \Sigma^* \text{ s.t. } \mathrm{Proj}_\Sigma(x) = w \,\}.$$

We note that an automaton $\mathcal{M}$ accepts a word $w \in \Sigma^*$ iff there is a path labeled exactly by symbols in w, which starts from any state in I and ends in any state in F.

We distinguished the set $\mathrm{FPath}(\mathcal{M})$ from paths in $\mathrm{Path}(\mathcal{M})$ as the set of all paths, which starts from any element in I and ends with a state in F, as the set of all **accepting paths** of $\mathcal{M}$. In particular, we define $\mathrm{FPath}(\mathcal{M})$ as follows,

DEFINITION 6.

$$\mathit{FPath}(\mathcal{M}) = \mathit{Path}(\mathcal{M}) \cap I \cdot (\Sigma \cdot Q)^* \cap (Q \cdot \Sigma)^* \cdot F.$$

THEOREM 1. *Let $\mathcal{M} \in$* **NFA**.

$$\mathrm{Proj}_\Sigma(\mathit{FPath}(\mathcal{M})) = L(\mathcal{M}).$$

PROOF. $\mathrm{FPath}(\mathcal{M})$ is the set of all strings of the form $v = q_0 a_1 q_1 a_2 q_2 \cdots a_n q_n$ for some n, where for all i, $1 \leq i \leq n$, $a_i \in \Sigma, q_i \in Q, q_0 \in I$, and $q_n \in F$. Since v represents a path in $\mathcal{M}$, $\forall i, 1 \leq i \leq n, (q_{i-1}, a_i, q_i) \in \Delta$ and thus in $\widehat{\Delta}$. Also, since $(q_0, a_1, q_1) \in \widehat{\Delta}$, and $(q_1, a_2, q_2) \in \widehat{\Delta}$, then $(q_0, a_1 a_2, q_2) \in \widehat{\Delta}$. Similarly, since $(q_2, a_3, q_3) \in \widehat{\Delta}$, then $(q_0, a_1 a_2 a_3, q_3) \in \widehat{\Delta}$. For the same reason, we will have $(q_0, a_1 a_2 \cdots a_m, q_m) \in \widehat{\Delta}$ for all $m \leq n$. Hence, $(q_0, a_1 a_2 \cdots a_n, q_n) \in \widehat{\Delta}$ and $\mathrm{Proj}_\Sigma(\mathrm{FPath}(\mathcal{M}))$ will be all strings $w = a_1 a_2 \cdots a_n$. Therefore for all w, $(q_0, w, q_n) \in \widehat{\Delta}$, and $\mathrm{Proj}_\Sigma(\mathrm{FPath}(\mathcal{M}))$ will be the set of all the strings accepted by $\mathcal{M}$ and thus $\mathrm{Proj}_\Sigma(\mathrm{FPath}(\mathcal{M})) = L(\mathcal{M})$. □

We shall call the set $\mathrm{FPath}(\mathcal{M})$ as a **path language** recognized by $\mathcal{M}$. We note that equivalent automata may recognize different path languages.

Since our model is nondeterministic, we can expect at least one computation which leads to an accepting state for all words found in the set of language accepted by some $\mathcal{M} \in$ **NFA**. We denote the set of successful paths that an automaton $\mathcal{M}$ made for an input w as $\mathrm{FPath}(\mathcal{M}(w))$.

3. THE COMPUTATION TREES

Let $\mathcal{M} = (Q, \Sigma, I, \Delta, F) \in$ **NFA**. We associate the following **ambiguity function:**

$$\mathrm{ambig}_\mathcal{M} : \Sigma^* \cup N \longrightarrow N,$$

such that for all $w \in \Sigma^*$,

$$\mathrm{ambig}_\mathcal{M}(w) = |\{\, u \in \mathrm{FPath}(\mathcal{M}) \mid \chi_\Sigma(u) = w \,\}|$$

and for all $n \in N$,

$$\mathrm{ambig}_\mathcal{M}(n) = \max_{w \in \Sigma^*;\, |w| \leq n} \mathrm{ambig}_\mathcal{M}(w).$$

It is imperative that, the above function measures the amount of paths which can be traced successfully from an element of I and ending with a state in F by an automaton on a given input of length at most n.

DEFINITION 7. *Let $\mathcal{M} = (Q, \Sigma, I, \Delta, F) \in$* **NFA** *and $w \in \Sigma^*$. The* **computation tree** *of w on $\mathcal{M}$ is defined by*

$$T_{w;\mathcal{M}} = \{\, x \in Q^* \mid \exists\, z \in \Sigma^* \quad \exists\, v \in \mathit{Path}(\mathcal{M}) \cap I \cdot (\Sigma \cdot Q)^*$$

$$s.t. \quad \chi_Q(v) = x \quad \wedge \quad \mathrm{Proj}_\Sigma(v) \cdot z = w \,\}.$$

A path $x \in T_{w;\mathcal{M}}$ is an **accepting path** *if and only if $|x| = |w| + 1$ and the last symbol of x is in F.*

We also note that, a computation tree of a word w on $\mathcal{M}$ is simply a proliferation of states on the said input by $\mathcal{M}$.

We can also associate the ambiguity of an input word w with the number of accepting leaves of its computation tree. That is, for any automaton $\mathcal{M}$ and any word $w \in \Sigma^*$,

$$\mathrm{ambig}_{\mathcal{M}}(w) = |\{\, x \in T_{w;\mathcal{M}} \mid x \text{ is an accepting path } \}|.$$

DEFINITION 8. *For all $w \in \Sigma^*$ and $\mathcal{M} \in UFA$, that is the* **class of unambiguous nondeterministic finite automata**, *whenever $ambig_{\mathcal{M}}(w) \leq 1$.*

LEMMA 1. *Let $\mathcal{M} \in$ **NFA**.For all$w \in \Sigma^*$,*

$$|FPath(\mathcal{M}(w))| \leq 1 \text{ if and only if } \mathcal{M} \in \mathbf{UFA}.$$

PROOF. Let $w \in \Sigma^*$, if $|\mathrm{FPath}(\mathcal{M}(w))| \leq 1$, then for each $w \in L(\mathcal{M})$, there is at most one accepting path, say $v = q_0 a_1 q_1 a_2 q_2 \cdots a_n q_n$ for some $n \in N$, where $q_0 \in I, q_n \in F$, such that $w = \mathrm{Proj}_\Sigma(v) = a_1 a_2 \cdots a_n$. Hence, there is at most one $x = \mathrm{Proj}_Q(v) = q_0 q_1 q_2 \cdots q_n$, where x is a sequence of states of the accepting path v. Then $\mathcal{M} \in UFA$.

The converse follows immediately from the definition of UFA. □

Intuitively, ambiguity of an $\mathcal{M} \in$ **NFA** connotes the number of ways it can accept an input $w \in L(\mathcal{M})$ or the cardinality of the set $\mathrm{FPath}(\mathcal{M})$.

4. EXAMPLES

Before we present an automaton that negates Strict Tree Property, we first give the following concepts. An automaton $\mathcal{M} \in$ **UFA** for language L is minimal iff it recognizes L and among its equivalent automaton in **UFA**, it has the minimum number of states. We note that such minimal automaton in **UFA** is not unique.

EXAMPLE 1. *Let $\mathcal{M} = (Q, \Sigma, \{q_0\}, \Delta, F) \in$ **NFA** s.t.*

$Q = \{\, q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7 \,\}$

$\Sigma = \{\, 0, 1 \,\}$

$F = \{\, q_3 \,\}$

$\Delta = \{\, (q_0, 0, q_0), (q_0, 1, q_0), (q_0, 1, q_1), (q_0, 1, q_4), (q_0, 0, q_6),$

$(q_1, 0, q_2), (q_2, 1, q_3), (q_4, 0, q_5), (q_5, 0, q_3), (q_6, 1, q_7),$

$(q_7, 0, q_3) \,\}.$

One can easily construct two different minimal automata in **UFA** *that accepts the same language as $\mathcal{M}$.*

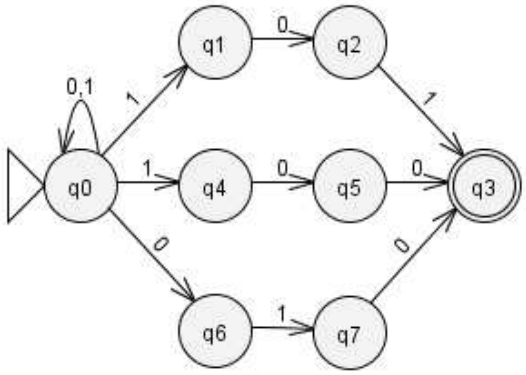

Figure 1: An NFA having more than one equivalent minimal automata accepting the same language.

Although, we have defined formally the above computation tree in our example, we will explicitly construct a computation tree of a particular input word by a minimal automaton below.

EXAMPLE 2. *Let $\mathcal{M}_1 = (Q, \Sigma, \{q_0\}, \Delta, F) \in$ **NFA** s.t.*

$Q = \{\, q_0, q_1, q_2, q_3, q_4, q_5 \,\}$

$\Sigma = \{\, 0, 1 \,\}$

$F = \{\, q_3 \,\}$

$\Delta = \{\, (q_0, 0, q_0), (q_0, 1, q_0), (q_0, 1, q_1), (q_0, 0, q_4), (q_1, 0, q_2),$

$(q_2, 0, q_3), (q_2, 1, q_3), (q_4, 1, q_5), (q_5, 0, q_3) \,\}.$

Notice that

$FPath(\mathcal{M}_1) =$

$$q_0\{\,\{0,1\}q_0\}^*\,\{\{1\,q_1\,0\,q_2\,\{0,1\}\,q_3\} \cup \{0\,q_4\,1\,q_5\,0\,q_3\}\,\}.$$

The language recognized by $\mathcal{M}_1$ is

$\mathrm{Proj}_\Sigma(FPath(\mathcal{M}_1)) = L(\mathcal{M}_1) = \{\, 0, 1 \,\}^*\{\{1\,0\{0,1\}\} \cup \{0\,1\,0\}\}$

*It is not hard to convince ourselves that $\mathcal{M}_1 \in$ **UFA**, and that it is minimal. Moreover, it could be verified that $\mathcal{M}_1$ follows the Strict Tree Property for any input $w \in \Sigma^*$.*

EXAMPLE 3. *Let $\mathcal{M}_1' = (Q, \Sigma, \{q_0\}, \Delta, F) \in$ **NFA** s.t.*

$Q = \{\, q_0, q_1, q_2, q_3, q_4, q_5 \,\}$

$\Sigma = \{\, 0, 1 \,\}$

$F = \{\, q_3 \,\}$

$\Delta = \{\, (q_0, 0, q_0), (q_0, 1, q_0), (q_0, 1, q_1), (q_0, 0, q_4), (q_1, 0, q_2),$

$(q_1, 0, q_5), (q_2, 1, q_3), (q_4, 1, q_5), (q_5, 0, q_3) \,\}.$

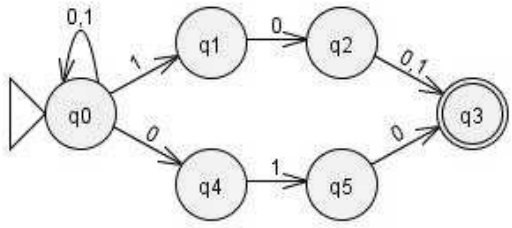

Figure 2: A minimal UFA that follows the Strict Tree Property.

It is not hard to see that the language recognized by $\mathcal{M}_1$ and $\mathcal{M}_1'$ is the same. The number of states of $\mathcal{M}_1'$ is equal to that of $\mathcal{M}_1$, hence, $\mathcal{M}_1'$ is also minimal.

$$FPath(\mathcal{M}_1) =$$

$$q_0\{\,\{0,1\}q_0\}^*\,\{\{1\,q_1\,\{0\,q_2\,1 \cup 0\,q_5\,0\}\,q_3\} \cup \{0\,q_4\,1\,q_5\,0\,q_3\}\,\}.$$

However, it is easy to see that the computation tree for the input 0101 by $\mathcal{M}_1'$ contradicts the Strict Tree Property.

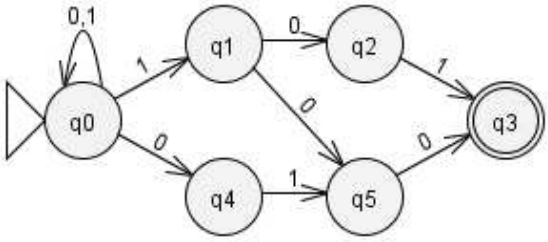

Figure 3: An unambiguous minimal automaton that contradicts the Strict Tree Property.

THEOREM 2. *[1] There exists a minimal automaton $\mathcal{M} \in$ **UFA**, which does not satisfy the Strict Tree Property.*

PROOF. Let $\mathcal{M} = \mathcal{M}_1'$, where $\mathcal{M}_1'$ is the automaton defined in the preceding example. □

We note that the automaton $\mathcal{M}_1'$ in the previous example is not deterministic at all. Therefore, we found a class of automata which is between the classes **DFA** and **UFA**, which satisfy the Strict Tree Property. Thus, we have the following corollary.

COROLLARY 1. *[1] Let* **StUFA** *be a class of automata which satisfies the Strict Tree Property. Then*

$$\textbf{DFA} \subset \textbf{StUFA} \subset \textbf{UFA}.$$

PROOF. This follows immediately from the preceding Lemma. □

5. STATE COMPLEXITY

In [1, 2], it was reported that **StUFA** are *'almost deterministic'*. However, we show below that the exponential gap between **NFA** and **DFA** with respect to their state complexities is also true between **StUFA** and **DFA**.

PROPOSITION 1. *Let L be a regular language accepted by $\mathcal{M} \in$ **StUFA**. Then $\exists\, \mathcal{A} \in$ **DFA**, where $L(\mathcal{A}) = L(\mathcal{M})$ and*

$$|Q_{\mathcal{A}}| \leq 2^{|Q_{\mathcal{M}}|},$$

where $Q_{\mathcal{A}}$ and $Q_{\mathcal{M}}$ are the finite set of states of $\mathcal{A}$ and $\mathcal{M}$, respectively.

PROOF. This follows from the fact that **StUFA** is a subset of **NFA.** □

COROLLARY 2. *There exists a regular language L_n accepted by some $\mathcal{M} \in$ **StUFA** such that no* **DFA** *can accept in less than $2^{|Q_{\mathcal{M}}|-1}$ states.*

PROOF. The language

$$L_n = \{\, x1y \,|\, x \in \{\,0,1\,\}^*,\, y\{\,0,1\,\}^{n-1}\,\}$$

witnesses this gap. □

For a given sequence of automata $(\mathcal{M}_i)_{i\geq 0}$, we define $|Q_{\mathcal{M}_i}|$ as the number of states of $\mathcal{M}_i$. For a function f on the set of natural numbers with $f \notin O(n)$ and two classes X_1 and X_2 of finite automata, we will say that there is a *$f(n)$-size gap* or *$f(n)$ separation* between X_1 and X_2, denoted by

$$X_1 \prec_{f(n)} X_2,$$

if there is a sequence of regular languages $L_i \in (2^{\Sigma^*})^N$ for $i \geq 0$, such that there exists a sequence $(\mathcal{M}_i)_{i\geq 0} \in X_2^N$ with $L(\mathcal{M}_i) = L_i$ and $|Q_{\mathcal{M}}| \in O(i)$ and all sequences $(\mathcal{N}_i)_{i\geq 0} \in X_1^N$ with $L(\mathcal{N}_i) = L_i$ satisfy $|Q_{\mathcal{N}}| \in \Omega(f(i))$.

Therefore, by Corollary 2, we have the following:

THEOREM 3.

$$\textbf{DFA} \prec_{2^{n-1}} \textbf{StUFA}.$$

It is still not known whether the gap between **StUFA** and **UFA** is exponential or not.

6. MAIN RESULTS

In this section we give necessary and sufficient condition for a minimal $\mathcal{M} \in$**UFA** to be in **StUFA.** We assume that all $\mathcal{M} \in$**UFA** below are minimal. First, we provide the following definitions.

DEFINITION 9. *Let $\mathcal{M} \in$ **UFA**. For all $w \in L(\mathcal{M})$.*

$$^1C_w \in T_{w;\mathcal{M}}$$

$$\textit{whenever } \exists\, v \in \mathit{Path}(\mathcal{M}) \cap I \cdot (\Sigma \cdot Q)^*, \mathrm{Proj}_{\Sigma}(v) = w$$

and $\text{Proj}_Q(v) = {}^1C_w$ *is an accepting path.*

All other paths which leads to either non-accepting or unfinished computation of $\mathcal{M}$ *on* w *will be denoted by* ${}^iC_w$*, for* $i \geq 2$.

DEFINITION 10. *For all* $w \in (\Sigma \cup Q)^*$*, we define the set of all proper leading symbols of* w *as its set of proper prefixes and is denoted by* $\mathbf{Pref}(w)$*. Note that* $\forall\, v \in \mathbf{Pref}(w)$, $|v| \leq |w| - 1$.

The following are easy observations.

OBSERVATION 1. *For all* $i \geq 2$,

$$\mathbf{Pref}\,({}^1C_w) \cap \mathbf{Pref}\,({}^iC_w) \neq \emptyset.$$

OBSERVATION 2. $\mathcal{M} \in \mathbf{UFA}$*, then* $\forall\, w \in L(\mathcal{M})$, $\forall\, i, j \geq 1$, $i \neq j$, $\exists\, {}^iC_w, {}^jC_w \in T_{w;\mathcal{M}}$ *s.t.*

$$\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \neq \emptyset.$$

LEMMA 2. *Let* $\mathcal{M} \in \mathbf{UFA}$.

$\mathcal{M} \in \mathbf{StUFA}$ *iff* $\forall\, i, j \geq 1$,

$$\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \subseteq \mathbf{Pref}\,({}^1C_w).$$

PROOF. ($\Rightarrow$) Let $\mathcal{M} \in \mathbf{StUFA}$. Let $w \in L(\mathcal{M})$. Suppose there are $i, j \geq 2$, such that $\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \not\subseteq \mathbf{Pref}\,({}^1C_w)$.
Therefore, we can find $u \in \mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w)$, s.t. $u \notin \mathbf{Pref}\,({}^1C_w)$. If we denote by $|w|$ the length of any string w, then it should be clear that $|u| < |{}^iC_w|, |{}^jC_w|$. Hence, there is a $v \leq^i C_w$, and a $v' \leq^j C_w$, such that $v = xy$, and $v' = xy'$, for some $y, y' \in Q^+$. This violates the strict tree property, and hence, $\mathcal{M} \notin \mathbf{StUFA}$. Therefore, for all i and j,

$$\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \subseteq \mathbf{Pref}\,({}^1C_w).$$

($\Leftarrow$) Let $\mathcal{M} \in \mathbf{UFA}$. Let for all $i, j \geq 1$,

$$\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \subseteq \mathbf{Pref}\,({}^1C_w).$$

We need to show that $\mathcal{M} \in \mathbf{StUFA}$. That is, it follows the strict tree property.

Suppose $\mathcal{M} \in \mathbf{UFA}$ but does not follow the strict tree property. This means there exists a nondeterministic compuatation path from ${}^1C_w$ that has another nondeterministic branch. Let such nondeterministic path be ${}^iC_w$, for some positive integer i. Let its nondeterministic computation path other than that with ${}^1C_w$ be denoted by ${}^jC_w$, for some positive integer j.

Now, there exists some $u \in \mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w)$, such that $u \notin \mathbf{Pref}\,({}^1C_w)$. Therefore, $\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) \not\subseteq \mathbf{Pref}\,({}^1C_w)$. But this is a contradiction. Therefore, $\mathcal{M} \in \mathbf{StUFA}$. □

THEOREM 4. *Let* $\mathcal{M} \in \mathbf{UFA}$*. For all* $w \in L(\mathcal{M})$,

$\mathcal{M} \in \mathbf{StUFA}$ *if and only if* $\forall\, i, j \in N$, $i, j \geq 1, i \neq j$,

$\exists\, S \subseteq \mathbf{Pref}\,({}^1C_w)$ *s.t.* $\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) - S = \emptyset$.

PROOF. ($\Rightarrow$) Let $\mathcal{M} \in \mathbf{StUFA}$. By Observation 2, we have for all $w \in L(\mathcal{M})$, and for all $i, j \geq 1$,

$$\mathbf{Pref}({}^iC_w) \cap \mathbf{Pref}({}^jC_w) \neq \emptyset.$$

Define

$$S_{ij} = \mathbf{Pref}({}^iC_w) \cap \mathbf{Pref}({}^jC_w).$$

Let

$$S = S_{ij} \cap \mathbf{Pref}\,({}^1C_w).$$

We need to show that $S_{ij} - S = \emptyset$.

Suppose $S_{ij} - S \neq \emptyset$. Then there is a $u \in S_{ij}$ such that $u \notin S$. That means, there is a $u \in S_{ij}$, for all $x \in S$, such that $|x| < |u| \leq |w|$, where $|x|, |u|$, and $|w|$ are lengths of strings x, u, and w, respectively.

Let $u = \max\{\, |v| \mid v \in S_{ij} \,\}$. In particular, define

$$u = q_0 q_1 q_2 \cdots q_{k-1} q_f,$$

where q_0 is the start state, q_f is some final state (not necessary accepting). Since $|u| < |{}^iC_w|, |{}^jC_w|$, then there is a $v \in \mathbf{Pref}\,({}^iC_w)$, and $v' \in \mathbf{Pref}\,({}^jC_w)$, such that $v = uy$, and $v' = uy'$, for some $y \in \mathbf{Pref}\,({}^iC_w)$, and $y' \in \mathbf{Pref}\,({}^jC_w)$. Therefore, this tells us that the computation tree of $\mathcal{M}$ for w does not follow the strict tree property , hence $\mathcal{M} \in \mathbf{StUFA}$. Therefore, $S_{ij} - S = \emptyset$.

($\Leftarrow$) This follows directly from Lemma 2. □

DEFINITION 11. *Let* $\mathcal{M} \in \mathbf{UFA}$*. Define*

$$S_{ij} = \mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w)$$

where the ${}^kC_w$*'s are as defined in Definition 9,* $k \in \{i, j\}$.

COROLLARY 3. *Let* $\mathcal{M} \in \mathbf{UFA}$.

$\mathcal{M} \in \mathbf{StUFA} \iff \forall\, w \in L(\mathcal{M})$, $\forall\, i, j \in N$, $i, j \geq 1, i \neq j$,

$\exists$ *an ordering of the* S_{ij}*'s s.t.* $S_{ij}^{(1)} \subseteq S_{ij}^{(2)} \subseteq \cdots \subseteq S_{ij}^{(n)} \subseteq \mathbf{Pref}\,({}^1C_w)$.

PROOF. ($\Rightarrow$) From Theorem 4, $\forall\, w \in L(\mathcal{M})$, $\forall\, i, j \in N$, $i, j \geq 1, i \neq j$, $\exists S \subseteq \mathbf{Pref}\,({}^1C_w)$ s.t. $\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) - S = \emptyset$

Note that $\forall i, j$, $S_{ij} = \mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w)$. By Lemma 2 each of the S_{ij} is contained in $\mathbf{Pref}\,({}^1C_w)$. Hence $\exists$ an ordering of the S_{ij}'s such that $S_{ij}^{(1)} \subseteq S_{ij}^{(2)} \subseteq \cdots \subseteq S_{ij}^{(n)} \subseteq \mathbf{Pref}\,({}^1C_w)$.

($\Leftarrow$) $\forall\, i, j \in N$, $i, j \geq 1, i \neq j$, $S_{ij} = \mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w)$

Applying Theorem 4, take $S = S_{ij}$. $\mathbf{Pref}\,({}^iC_w) \cap \mathbf{Pref}\,({}^jC_w) - S_{ij} = \emptyset$. Hence, $\mathcal{M} \in \mathbf{StUFA}$. □

Acknowledgements

The authors would like to thank the people from the Algorithms and Complexity Lab of the Department of Computer Science, UPDiliman for the support they provided.

7. REFERENCES

[1] H.N. Adorna. On the Computation Tree of an Unambiguous Nondeterministic Finite Automaton, *In: Abstract Proceeding of the Annual Convention of the Mathematical Society of the Philippines,* ADMU, Quezon City, May 2002.

[2] H.N. Adorna,: Some Problems in Descriptional Complexity of Finite Automata. *Proc. 5th Phil. Computing Science Congress* 2005, pp 27-32.

[3] J.E. Hopcroft, J.D. Ullman: *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, Reading, MA, 1979

[4] J. Hromkovič, J. Karhumäki, H. Klauck, G. Schnitger, S. Seibert: Communication complexity method for measuring nondeterminism in finite automata. *Information and Computation* 172, 2002, pp 202-217.