Characteristic Sets for Learning k-Acceptable Languages

Anuchit Jitpattanakul[1] and **Athasit Surarerks**[2], Non-members

ABSTRACT

Learnability of languages is a challenging problem in the domain of formal language identification. It is known that the efficiency of a learning technique can be measured by the size of some good samples (representative or distinctive samples) formally called a characteristic set. Our research focuses on the characteristic set of k-acceptable languages. We proposed a Gold-style learning algorithm called *KRPNI* which applied the grammatical inference technique to identify a language and expressed it by a k-DFA. In this paper, we study the existence of such characteristic sets. Our theoretical results show that there exists a polynomial characteristic set for a k-acceptable language. It is found that the size of the characteristic set depends on the value of k, instead of the size of an alphabet.

Keywords: Learnability, k-Acceptable Languages

1. INTRODUCTION

Grammatical inference is considered as a model for identifying the characteristic of a language or words in the languages. Typically, the grammatical representation mostly refers an abstract machine named a *finite state automaton* or a set of grammatical rules called a *grammar*.

The study on learnability of classes of formal languages has received much attention in grammatical inference research field. When the learnability of some classes is proved, this confides that there exists at least one learning algorithm that can return a correct hypothesis when samples are available enough. There are a number of researches that take usefulness of knowing the learnability of some classes. These works have applied learning algorithms to various practical problems. For example, *RPNI* algorithm was applied to music style recognition, see detail in [1].

Theoretically, there have been a number of published papers providing proofs on learnability of both trivial and nontrivial classes of languages [2]. Learnability of classes of languages has been considerably interested for researchers since Gold's works was published in 1967 [3]. In the seminal work, Gold introduced an explanatory learning model called identification in the limit and used this model to study on learnability of nontrivial classes of languages in Chomsky's hierarchy. One of the interesting results was shown that a recursively enumerable language is learnable from examples that are labelled whether they are strings of the target language or not. The examples, which are used in learning context, are called positive examples if they are in the language and are called negative examples if they are not so.

The Gold's learning model only convinces us that the correct grammatical representation would be eventually identified, but the issue of complexity of learning is regardless. To concern this important issue, Gold [4] further studied the learnability and proposed another model by additionally concerning complexity in the process of language learning. This learning model is *called identification in the limit from polynomial time and data*. Two conditions, to prove that any class is efficiently learnable, are these followings: a learner must identify a representation with polynomial time of size of examples and a learner must correctly identify a representation of the target language when a characteristic set is given. With the best knowledge of authors, the informal notion of the characteristic set was first introduced in his article. Gold also proved in his work that a class of regular languages is identifiable in the limit from polynomial time and data.

A formal notion of a characteristic sets was defined by Higuera in [5]. In this work, Higuera also gave a formal definition of identification in the limit from polynomial time and data. Most of research in grammatical inference adopts this formal definition as a fundamental definition for proving efficient learnability. A number of classes of languages have been shown that they are identifiable in the limit from polynomial time and data such as the following results: a class of regular languages by Oncina and Garcia [6], a class of commutative regular languages by Gomez and Alvarez [7], a class of one-clock deterministic timed automata by Verwer et al. [8], a class of deterministic linear languages by Higuera [9], and a class of strictly deterministic finite automata [10].

A k-acceptable language is a formal language which is recognized by a k-edge deterministic finite automaton (k-DFA) introduced in [11]. The problem of learnability of these was classes proposed as

Manuscript received on July 30, 2010 ; revised on July 25, 2011.

[1,2] The authors are with Engineering Laboratory in Theoretical Enumerable System Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Patumwan, Bangkok, Thailand 10330 , E-mail:januchit@yahoo.com and athasit@cp.eng.chula.ac.th

an open problem in the paper. By our previous work [12], the learning algorithm $KRPNI$ was proposed to learn k-acceptable language. Therefore, in this paper, we study on existence of characteristic set for k-acceptable languages. We also prove that the size of characteristic examples for each k-acceptable language depends on the value of k, instead of the size of alphabet.

The remains of this paper are organized as follows. Section 2 provides related definitions and notations of language learning. In section 3, we give a learning algorithm $KRPNI$ for k-acceptable languages. The characteristic sets for learning k-acceptable languages are described in section 4. The theoretical results are also shown in this section. Section 5 dedicates to discuss the theoretical results and actual languages. Finally, the last section provides the conclusion and future works.

2. PRELIMINARIES

In this section we give some basic definitions and notations concerning languages and their machines, and also the notion of characteristic set.

2.1 Basic Definitions and Notations

An *alphabet* Σ is a finite set of symbols called letters. A finite sequence of letters from Σ is called a *string*. Let w be a string and the length of strings is denoted by $|w|$. The string with length zero is called the *nullstring* denoted by λ. The infinite set of all strings over Σ is denoted by Σ^*. Given a concatenated string $w = uv$, a string u is a $prefix$ of w if and only if there exists a string v in Σ^*. In this work, we define an $orderalphabet\Sigma_{\leq}$ as a poset, where Σ is an alphabet and $\leq$ is a partial order over Σ. To order strings, we use the lexicographic-length order over Σ^* defined by $\forall u, v \in \Sigma^*, u < v$ if and only if $|u| < |v|$ or there exists $w, u^{'}, v^{'} \in \Sigma^*$ and two letters $x < y \in \Sigma$ such that $u = wxu^{'}$, $v = wyv^{'}$. A language L defined over Σ is a subset of Σ^*. The complement of L is defined by $L^{'} = \Sigma^* - L$. Given $u, v \in \Sigma^*$, we define the prefix set of a set $S \subseteq \Sigma^*$ as $Pref(S) = u \in \Sigma^* : uv \in S$. Similarly the prefix set of L is defined as $Pref(L) = \{u \in \Sigma^* : uv \in L\}$.

A *deterministic finite automaton* (DFA) is a 6-tuple $M = (\Sigma, Q, q_0, F_A, F_R, \delta)$ where Σ is a finite alphabet, Q is a finite set of states, q_0 is the initial state, $F_A, F_R \subseteq Q$ are a set of accepting states and a set of rejecting states respectively, and δ is the transition function defined as $\delta : Q \times \Sigma \to Q$ such that $|\delta(q, a)| \leq 1$ for each $q \in Q$ and for each $a \in \Sigma$. The transition function can be extended to a mapping $\delta^* : Q \times \Sigma^* \to Q$ in the following inductive way: $(i)\delta^*(q, \lambda) = q$, for each state $q \in Q$, and $(ii)\delta^*(q, wa) = \delta\ (\delta^*(q, w), a)$, for each $q \in Q$, for each $a \in \Sigma$, and for each $w \in \Sigma^*$. A language associated with the DFA is the set of strings that can be recognized by a DFA which is called a *regular language*.

Let $\mathbf{L}$ be a class of languages and $\mathbf{M}$ be a class of grammatical representations for $\mathbf{L}$. A language recognized by $\mathbf{M}$is defined as $L(M) = \{w \in \Sigma^* : \delta^*(q_0, w)F\}$. The size of a language $L \in \mathbf{L}$ is the size of the smallest $M \in \mathbf{M}$ of the considered class. It is defined as $|L| = \min\{\|M\| : \mathbf{L}(M) = L$ for each $M \in \mathbf{M}\}$. The size of an automaton $\|M\|$ is defined, in this article, as a number of states. That is $\|M\| = |Q|$.

2.2 k-Acceptable languages and k-edge deterministic finite automata

A k-edge deterministic finite automaton (k-DFA) is a 6-tuple $M = (\Sigma_{\leq}, Q, q_0, F_A, F_R, \delta)$ where $\Sigma_{\leq}$ is a finite ordered alphabet, Q is a finite set of states, q_0 is the initial state, $F_A \subseteq Q$ is a set of accepting states and $F_R \subseteq Q$ is a set of accepting states, $\delta : Q \times \Sigma_{\leq} \times \Sigma_{\leq} \to Q$ is the transition function defined as $\forall q \in Q, |\{x, y : \delta(q, x, y) \neq \varnothing\}| \leq k$, and if $\delta(q, a_1, b_1) \neq \delta(q, a_2, b_2)$ then $\{z : a_1 \leq z \leq b_1\} \cap \{z : a_2 \leq z \leq b_2\} = \varnothing$. For each transitions $\delta(q, a, b)$, we define the *lower bound* and the *upper bound* of each transitions as $Lb(\delta(q, a, b)) = a$, $Ub(\delta(q, a, b)) = b$, respectively. The extended transition function $\delta^* : Q \times \Sigma^*_{\leq} \to Q$ is defined as $\delta^*(q, \lambda)$ and $\delta^*(q, aw) = \delta^*(q^{'}, w)$where $x \leq a \leq y$ and $\delta(q, x, y) = q^{'}$ such that $q, q^{'} \in Qa, x, y \in \Sigma_{\leq}, w \in \Sigma^*_{\leq}$. The languages recognized by a k-DFA are called $k - acceptablelanguages$.

A canonical $k-edgedeterministicfiniteautomaton$ of a language L denoted by M_c is a homomorphic image of every k-DFA recognizing L. Let I_L be indistinguishability relation on $\Sigma^*_{\leq}$. We define the canonical $k - edge$ deterministic finite automata of L as $M_c = (\sum_{\leq}, Q_c, q_0, F_{Ac}, F_{Rc}, \delta_c)$,where $Q_c = \{[q_u]_{\sim L} : u \in \sum^*\}$, $q_0 = [q_\lambda]_{\sim L}$, $F_{Ac} = \{[q_u]_{\sim L} : u \in L\}$, $F_{Rc} = \{[q_u]_{\sim L} : u \in L\}$, $\delta_c([q_u]_{\sim L}, a, b) = [q_{uz}]_{\sim L}$ such that $a \leq z \leq b$.

Example
The finite automaton from Fig. 1 recognizes language $((1+2+3)+(4+5)1^*(2+3+4+5))^*$. This automaton is 2-DFA because $\forall q \in Q, |\{(x, y) : \delta(q, x, y) \neq \varnothing\}| \leq 2$ and for state $q_0 : \delta(q_0, 1, 3) \neq \delta(q_0, 4, 5) \neq \varnothing$ then $\{z : 1 \leq z \leq 3\} \cap \{z : 4 \leq z \leq 5\} = \varnothing$ for state $q_3 : (q_3, 1, 1) \neq (q_3, 2, 5) \neq \varnothing$ then $\{z : 1 \leq z \leq 1\} \cap \{z : 2 \leq z \leq 5\} = \varnothing$. Notice that the same language can also be recognized by a 3-DFA, but not by a 1-DFA.

2.3 Characteristic sets for language learning

Let L be a target language in a class $\boldsymbol{L}$. With context of Gold-style leaning, strings in Σ^* are sampled as examples for learning. A string $w \in \Sigma^*$ is called a positive example if $w \in L$ and it is called a

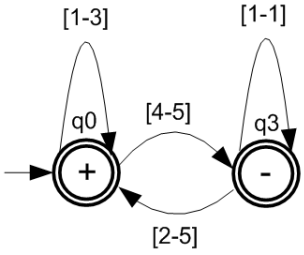

Fig.1: *2-DFA*

negative example if not so. In the process of learning, the main purpose is to construct an equivalent grammatical representation by using an input sample $S = (S+, S-)$ for L, where $S+ \subseteq L$ is a set of positive examples and $S- \subseteq \Sigma^* - L$ is a set of negative examples. We here modify the set-inclusion operators for input samples such that they operate on the set of positive examples and the set of negative examples separately, i.e., if $S = (S+, S-)$ and $S = (S+, S-)$ then SS means $S+S+$ and $S-S-$. The size of sample $S = (S+, S-)$ is defined as $||S|| = \Sigma_i |w_i|$ such that $w_i \in S+ \cup S-|$.

A learning algorithm $\boldsymbol{A}$ is a mapping function defined as $\boldsymbol{A}$:$\boldsymbol{S} \rightarrow \boldsymbol{M}$ such that $\boldsymbol{S}$ is a set of all samples which is used for learning any language L in language class $\boldsymbol{L}$ by identifying a grammatical representation M in corresponding grammatical representation class $\boldsymbol{M}$. We say that the algorithm $\boldsymbol{A}$ *converges* to $M \in \boldsymbol{M}$ from $S \in \boldsymbol{S}$ if and only if $\boldsymbol{A}(S) = M$ such that $\mathbf{L}(M) = L$. Given a language L, the algorithm $\boldsymbol{A}$ *identifies* L if $\boldsymbol{A}$ *converges* to $M \in \boldsymbol{M}$ for any $S \in \boldsymbol{S}$ and $S \subseteq L$. An algorithm $\boldsymbol{A}$ is said to efficiently identify M in the limit if $\boldsymbol{A}$ requires time polynomial in the size of any input sample S for M, and if A is given a set covering a characteristic set CS for M then $\boldsymbol{A}$ converge to M such that the size of CS is bounded by a polynomial in the size of M.

The formal definition of the characteristic set [5] is below.

Definition 1 (characteristic set)
A set of positive examples and negative examples is said to be a characteristic set CS = (CS+, CS-) of a target language L for a learning algorithm $\boldsymbol{A}$ *if CS satisfies these following conditions:*
1. Given an input sample CS, the learning algorithm $\boldsymbol{A}$ *returns a grammatical representation* $M \in \boldsymbol{M}$ *such that* $\boldsymbol{L}(M)=L$.
2. Given any input sample $S \supseteq CS$, *the learning algorithm* $\boldsymbol{A}$ *returns a grammatical representation M M such that* $\boldsymbol{L}(M) = L$.

3. LEARNING K-ACCEPTABLE LANGUAGES

In this section, we show a learning algorithm called $KRPNI$ proposed in our previous work [12]. The algorithm is used to learn a target k-acceptable language by identifying a k-DFA. The language information provided to the algorithm is a set of positive and negative examples. The algorithm always returns a k-DFA consistent with given positive and negative examples. The algorithm is shown in Fig. 2.

For the input set $S = (S+, S-)$, the algorithm $KRPNI$ initials with constructing a prefix tree automaton PTA(S) for the input sample S. Pairs of states will be chosen by lexicographic order. The process of merging may return a temporary automaton which is nondeterministic. To avoid the nondeterministic automaton, the merging needs to be recursively continued by maintaining the tree invariant property (The property of tree invariant is a sufficient condition for the determinization process to be finite). Then the obtained automaton will be checked consistency with the sample. In the best cases, it a characteristic set of the language includes in the input sample then the learning algorithm returns the k-DFA which is isomorphic to the target.

The algorithm $KRPNI$ composes of four main sub-procedures named *Choose*, *Secure*, *Compatible* and $Merge$.

- The function *Choose* returns a smallest state by considering lexicographic-length order of strings.
- The function *Secure* returns a k-DFA which a given state q_α and all its previous states of q_α are secure.
- The function *Compatible* returns a logical value. The value "True" is returned if a k-DFA is consistent with a given input sample $S = (S+, S-)$ and vice versa.
- The function $Merge$ returns a k-DFA which the state q_β have been recursively merged into the state q_ω .

Example
We demonstrate the running of the KPRNI algorithm on the task of learning k-DFA illustrated by Fig. 1. Assume that a sample $S = (S+, S-)$ where $S+ = \{3, 12, 42, 422\}$ be a set of positive examples and let $S- = \{4, 41\}$ be a set of negative examples over the ordered alphabet $\Sigma_{\leq} = \{1, 2, 3, 4, 5\}$. The automaton M=PTA(S) is depicted in Fig. 3a, where the number of states are equal to the cardinality of $Pref(S)$.

In the while loop, the algorithm chooses the state q_0 as the smallest state in the first round of iteration. Then, all children states of q_0 are sequentially merged together in order to make q_0 be a secured state. This step is shown in Fig. 3b. Next the children of q_0 are recursively merged into some previous states that compatibility constraint is true. We show this merging in Fig. 3c. The algorithm iteratively performs until all state is considered. Finally the result of 2-DFA is shown in Fig. 3d.

==========

Algorithm: *KRPNI*

Input: positive and negative samples $S = (S+, S-)$

Output: a k-DFA $M = (\Sigma_{\leq}, Q, q_0, F_A, F_R, \delta)$

$M \leftarrow$ PTA(S)

$K \leftarrow \emptyset$

While $Q\text{-}K \neq \emptyset$ **do**

$q_\alpha \leftarrow Choose(Q\text{-}K)$

$K \leftarrow K \cup \{q_\alpha\}$

$M \leftarrow Secure(M, q_\alpha)$

$B \leftarrow \{ q_\beta : q_\beta \in \delta(q_\alpha, a, b) : a, b \in \Sigma_{\leq} \}$

For $q_\beta \in B$ **do**

For $q_\omega \in K$ **do**

If $Compatible(Merge(M, q_\omega, q_\beta), S)$ **then**

$M \leftarrow Merge(M, q_\omega, q_\beta)$

Endif

Endfor

$K \leftarrow K \cup \{q_\beta\}$

Endfor

Endwhile

Return M

==========

Fig.2: *KRPNI Algorithm*

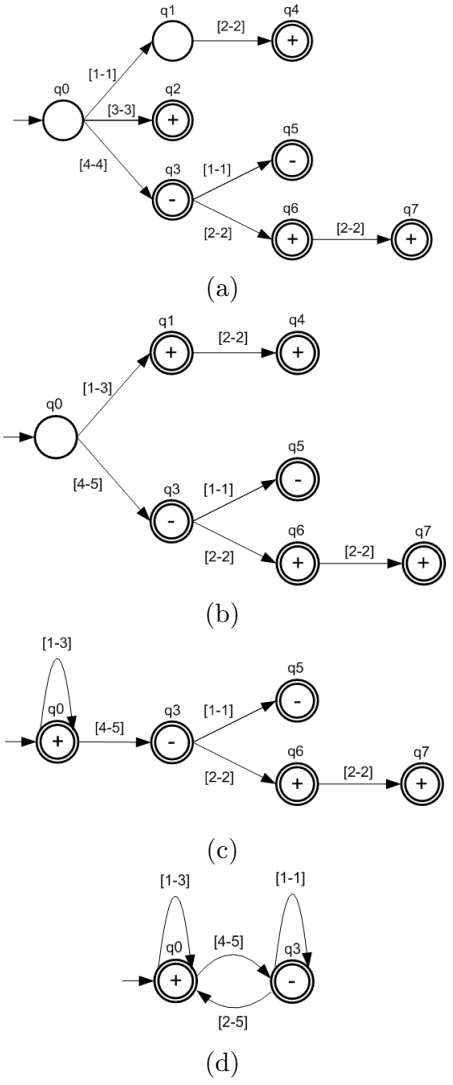

Fig.3: *Shows some steps of merging states for learning k-DFA*

4. CHARACTERISTIC SETS FOR LEARNING K-ACCEPTABLE LANGUAGES

In this section, we show the existence of a characteristic set $CS = (CS+, CS-)$ of k-acceptable languages L for $KRPNI$. The CS ensure the $KRPNI$ algorithm will return a k-DFA $M = (\Sigma_{\leq}, Q, q_0, F_A, F_R, \delta)$ such that $\mathbf{L}(M) = L$.

To construct a characteristic set, we need to define the *shortprefix* of a state $q \in Q$ denoted by $Short(q)$, the set of short prefixes of L denoted by $SP(\mathbf{L}(M))$, and the *kernelset* of L denoted by $N(\mathbf{L}(M))$ as follows:

- $Short(q) = \min\{u \in \Sigma^* : \delta^*(q_\lambda, u) = q\}$,
- $SP(\mathbf{L}(M))$=$u \in \Sigma^* : \forall qQ, u = Short(q)$,
- $N(\mathbf{L}(M)) = \{\lambda\} \cup \{uz : \forall q \in Q, u = Short(q), z = Lb(\delta(q, a, b))\}\{uz : \forall qQ, u = Short(q), z = Ub(\delta(q, a, b))\}$.

A set $CS = (CS+, CS-)$ is a characteristic set of $\mathbf{L}(M)$ for the algorithm $KPRNI$ if it satisfies the following conditions:
- $\forall u \in N(\mathbf{L}(M))$, if $\delta^*(q_0, u) \in F_A$ then $u \in CS+$ and if $\delta^*(q_0, u) \in F_R$ then $u \in CS-$,
- $\forall u \in\in SP(\mathbf{L}(M))$, $\forall v \in N(\mathbf{L}(M))$, if $\delta^*(q_0, u) \neq \delta^*(q_0, v)$ then $uw \in CS+$ and $vw \in CS-$ or $vw \in CS+$ and $uw \in CS-$, where w is a distinguishing string formally defined as $w = \min\{w \in \Sigma^* : (\delta^*(q_u, w) \in F_A \wedge \delta^*(q_v, w) \in F_R) \vee (\delta^*(q_u, w) \in F_R \wedge \delta^*(q_v, w) \in F_A)\}$.

Example

Construct a characteristic set $CS = (CS+, CS-) for k-DFA M = (\Sigma_{\leq}, Q, q_0, F, F_R, \delta) in Fig.1$.

Solution For each $q \in Q$, the short prefixes of state q_0 and state q_3 are "λ" and "4", respectively.

We construct the set of short prefixes of L recognized by M as

$$SP(\mathbf{L})(M) = \{\lambda, 4\}.$$

Then, we construct the kernel set of L from $SP(\mathbf{L}(M))$. So we have

$$N(\mathbf{L}(M)) = \{\lambda, 1, 3, 4, 5, 41, 42, 45\}.$$

Finally, we have $CS(M) = (CS+, CS-)$ such that

$$CS+ = \{\lambda, 1, 3, 42, 45\} \text{ and } CS- = \{4, 5, 41\}.$$

Let **A** denote the $KRPNI$ algorithm that returns k-DFA from a given input set $S = (S+, S-)$ and $CS = (CS+, CS-)$ be a characteristic set of a target k-acceptable language L recognized by k-DFA M. From the definition of the characteristic set (definition 1), we now must show the following lemmas:

Lemma 1. *Given a characteristic set $CS = (CS+, CS-)$ of a k-acceptable language L, the KRPNI algorithm returns a k-DFA M such that $\boldsymbol{L}(M) \subseteq L$.*

Proof : (By induction) Let L be a target k-acceptable language on an ordered alphabet $\Sigma_{\leq}$, and we denote two k-DFAs instep of proving as follows:
- $M = (\Sigma_{\leq}, Q, q_0, F_A, F_R, \delta)$ is a canonical k-DFA for a target L.
- $M_i = (\Sigma_{\leq}, Q_i, q_0, F_{Ai}, F_{Ri}, \delta_i)$ is a k-DFA built from $KRPNI$ algorithm at ith iteration.

The idea for proving this lemma is to show that $\mathbf{L}(\mathbf{A}(CS)) \in L(M)$. To do so, we have to show that the automaton M_i, that we build at any moment, is a subautomaton of M for a target language L.

We now define a homomorphism $f : Q_i \rightarrow Q$ *as* $f(q) = \delta^*(q_0, u_q)$ for each $q \in Q_i$, where u_q is a prefix of strings in the given CS such that the number of transitions in $\delta^* i(q_0, u_q) = q$ is minimum. So, we will show that M_i is homomorphic to a subautomaton of M by showing the following conditions hold:

(i) if q is in Q_i, then $f(q)$ is in Q,
(ii) if q is in F_{Ai}, then $f(q)$ is in F_A,
(iii) if q is in F_{Ri}, then $f(q)$ is in F_R,
(iv) for each $q \in Q_i$, and for each $a \in \Sigma_{\leq}$,
if $\delta_i(q, a, b) = p$, then $f(\delta_i(q, a, b)) = \delta(f(q), a, b)$.

Basis step: [show $\mathbf{L}(M_0) \subseteq \mathbf{L}(M)$]
As $KRPNI$ initiates with building a prefix tree acceptor for CS, we have that $M0 = PTA(CS) = (\Sigma_{\leq}, Q_0, q_0, F_{A0}, F_{R0}, \delta_0)$, where $q_0 = q, Q_0 = \{u_u : u \in Pref(CS)\}$, $F_{A0} = q_u \in Q_0 : u \in CS+, F_{R0} = q_u \in Q_0 : u \in CS-, \delta_0(q_u, a, a) = q_{ua}$ such that $u, ua \in Pref(CS)$.

To show (i), we suppose $q_u \in Q_0$. Since $u \in Pref(CS) \subseteq Pref(L(M))$ and $\delta^*(q_0, u) \in Q$, so we have that $f(q_u) \in Q$ by the defined homomorphism.

To show (ii and iii), we suppose $q_u \in F_{A0}$ and then we have that $u \in CS+ \subseteq \mathbf{L}(M)$. Thus, $\delta^*(q_0, u) \in F_A$ implies that $f(q_u) \in F_A$ by definition. Similarly we suppose $q_u \in F_{R0}$ and then we have that $u \in CS- \subseteq \Sigma^*_{\leq} - L(M)$. Thus, $\delta^*(q_0, u) \in F_R$ implies that $f(q_u) \in F_R$ by definition.

To complete proving in basis step, we will show (iv) hold. For$q, p \in Q_0$ such that $\delta_0(q, a, b) = p$, we let $u_p = u_q z : a \leq z \leq b$. From the manner of choosing u_p in PTA(CS), when u_q is a prefix of string in CS such that the number of transition in $\delta^*_0(q_0, u_q) = q$ is minimum, then $\delta^*(q_0, u_p) = p$ is minimum as well. Since $CS \subseteq \mathbf{L}(M)$, so we have that $f(p) = \delta^*(q_0, u_p)$.

R.H.S. $= \delta(f(q), a, b)$
$= \delta(\delta^*(q_0, u_q), a, b)$; by definition
$= \delta^*(q_0, u_q z)$; $a \leq z \leq b$
$= \delta^*(q_0, u_p)$; $u_q z = u_p$
$= f(p)$; by definition
$= f(\delta_0(q, a, b))$; $\delta_0(q, a, b) = p$
$=$ L.H.S

Therefore, we have $f(\delta_0(q, a, b)) = \delta(f(q), a, b)$.

The above proving (i), (ii), (iii) and (iv) imply that M_0 is homomorphic to a subautomaton of the canonical k-DFA M. Thus, $\mathbf{L}(M_0)$ $\mathbf{L}(M)$ is true.

Inductive step: [show if $\mathbf{L}(M_t) \subseteq \mathbf{L}(M)$ then $\mathbf{L}(M_{t+1}) \subseteq \mathbf{L}(M)$]

In this step of proving, we suppose this lemma holds for M_t and we show that it also holds for M_{t+1}. From $KRPNI$ algorithm, M_{t+1} is derived from merging states in M_t. Thus, we distinguish possible state merging into three cases.

Case 1: $[M_{(t+1)} = Merge(M_t, q_\omega, q_\beta)$ such that $\delta_t(q_u, a_1, b_1) = q_\beta$ and $\delta_t(q_u, a_2, b_2) = q_\beta]$. From merging with this case, we have $Q_{(t+1)} = Q_t\{q_\beta\} \subseteq Q_t$, $F_{A(t+1)} = F_{A(t)} - \{q_\beta\} \subseteq F_{A(t)}$, $FR(t+1) = F_{R(t)}$-$\{q_\beta\} \subseteq F_{R(t)}$, and $\delta_{(t+1)} = \delta_t - (\{(q_u, a, b, q_\beta)\} \cup \{(q_\beta, a, b, q_v) : q_v \in Q_t\}) \subseteq \delta_t$. By supposition, it is obvious that the conditions (i), (ii), (iii) and (iv) holds for $M_{(t+1)}$.

Case 2: $[M_{(t+1)} = Merge(M_t, q_\omega, q_\beta)$ such that $\delta_t(q_u, a_1, b_1) = q_\omega$ and $\delta_t(q_v, a_2, b_2) = q_\beta, q_u \neq q_v]$. From merging with this case, we have $Q_{(t+1)} = Q_t - \{q_\beta\} \subseteq Q_t, F_{A(t+1)} = F_{A(t)} - \{q_\beta\} F_{A(t)}, F_{R(t+1)} = F_{R(t)} - \{q_\beta\} \subseteq F_{R(t)}$, and $\delta_{(t+1)} = \delta_t - (\{(q_v, a, b, q_\beta)\} \cup \{(q_\beta, a, b, q_v) : q_v \in Q_t\}) \cup \{(q_v, a, b, q_\omega)\} \subseteq \delta_t$. Thus, it is obvious that the conditions (i), (ii), (iii) and (iv) holds for $M_{(t+1)}$ by supposition for M_t.

Case 3: $[M_{(t+1)} = M_t$ because $Merge(M_t, q_\omega, q_\beta)$ fails]. From merging with this case, we have $Q_{(t+1)} = Q_t, F_{A(t+1)} = F_{A(t)}, F_{R(t+1)} = F_{R(t)}$, and $\delta_{(t+1)} = \delta_t$. Thus, it is obvious that the conditions (i), (ii), (iii) and (iv) holds for $M_{(t+1)}$ by supposition for M_t. With three cases of the merging, we have that the statement is true. Thus, it follows that this lemma is true.

Lemma 2. *Given a characteristic set $CS = (CS+, CS-)$ of a k-acceptable language L, the KRPNI algorithm returns a k-DFA M_n such that $L \subseteq \boldsymbol{L}(M_n)$.*

Proof : To prove this lemma, we will show that $\mathbf{L}(M)$ $\mathbf{L}(\mathbf{A}(CS))$. We let $M_n = (\Sigma_{\leq}, Q_n, q_0, F_{An}, F_{Rn}, \delta_n)$ be k-DFA returned from $KRPNI$, i.e. $\boldsymbol{A}(CS) = M_n$. That means we must show that M is a subau-

tomaton of M_n.

Prove (i): [If q is in Q, then $f(q)$ is in Q_n] We suppose that $q \in Q$. If $q \in FA$ then clearly $u_q \in CS+$ by definition of the characteristic sample. It follows that u_q $\mathbf{L}(M_n)$. Hence, $\delta^*(q_0, u_q) \in Q_n$. It follows that $f(q) \in Q_n$ because of $f(q) = \delta^*(q_0, u)$ by definition. If $q \in F_R$ then clearly $uq \in CS-$ by definition of the characteristic sample. It follows that u_q Σ^* $\mathbf{L}(M_n)$. Hence,$\delta^*(q_0, u_q) \in Q_n$. If follows that $f(q) \in Qn$ because of $f(q) = \delta^*(q_0, u)$ by definition. Therefore, if q is in Q, then $f(q)$ is in Q_n.

Prove (ii): [If q is in F_A, then $f(q)$ is in F_{An}] We suppose q is in F_A. It follows that $u_q \in CS+$ by definition of the characteristic sample. Then, we have $\delta^*(q_0, u_q) \in F_{An}$ by manner of $KRPNI$. It follows that $f(q)F_{An}$. Therefore, if q is in F_A, then $f(q)$ is in F_{An}.

Prove (iii): [If q is in F_R, then $f(q)$ is in F_{Rn}] We suppose q is in F_R. It follows that $u_q \in CS-$ by definition of the characteristic sample. Then, we have $\delta^*(q_0, u_q) \in F_{Rn}$ by manner of $KRPNI$. It follows that $f(q) \in F_{Rn}$. Therefore, if q is in F_R, then $f(q)$ is in F_{Rn}. Prove (iv): [For every $q, p \in Q$, and every $a \in \Sigma_{\leq}$ such that $\delta(q, a, b) = p, f(\delta(q, a, b)) = \delta_n(f(q), a, b))$]

$$\begin{aligned}
\text{R.H.S.} &= \delta_n(f(q), a, b)) \\
&= \delta_n(\delta_n^*(q_0, u_q), a, b)) \text{ ; by definition} \\
&= \delta_n^*(q_0, u_q z); a \leq z \leq b \text{ by definition} \\
&= \delta_n^*(q_0, u_p); u_q z = u_p \\
&= f(p) \text{ ; by definition} \\
&= f(\delta(q, a, b)); \delta(q, a, b) = p \\
&= \text{L.H.S.}
\end{aligned}$$

As (i), (ii), (iii), and (iv) hold, so this lemma is true.

Lemma 3. *If given any input sample S including the characteristic set CS = (CS+, CS-) of a k-acceptable language L, then KRPNI algorithm returns a k-DFA M such that L(M) = L.*

Proof: It is obvious that the proof is obtained from Lemma 1 and Lemma 2.

Theorem 1. *There exists a characteristic set $CS = (CS+, CS-)$ of k-acceptable languages for $KRPNI$ algorithm.*

Proof: By Lemma 1, Lemma 2 and Lemma 3, we conclude by using definition 1 that there exists a characteristic set of k-acceptable language for $KRPNI$ algorithm.

Theorem 2. *The size of characteristic sets for k-acceptable languages is $O(n^3k)$ where n is size of k-DFA recognizing the language.*

Proof: By constructing of the characteristic set, the size of short prefix set $|SP(\mathbf{L}(M))$— is as many as the number of states of the canonical k-DFA $M = (\Sigma_{\leq}, Q, q_0, F_A, F_R, \delta)$. Suppose $|Q| = n$, so we have that $|SP(\mathbf{L}(M))| = n$. It clear that the number of strings in N(L(M)) is $n \cdot 2k+1$. Therefore the number of strings in $CS = (CS+, CS-)$ is $|CS+| < 2n^2 \cdot k$ and $|CS-| < 2n^2 \cdot k$. In the worst case, the maximum length of strings in $SP(\mathbf{L}(M))$ is equal to n and $N(\mathbf{L}(M))$ is equal to $n+1$. For the maximum length of a distinguishing string w is equal n. By considering the size of the strings in CS, we have that the possible length of strings in CS is less than $2n+1$. Thus, it show that $||CS|| \in O(n^3k)$.

5. DISCUSSION

The theoretical results can be implied for any formal languages defined over ordered alphabet. The family of music languages is classified in this context because their alphabets are naturally ordered in some modes. To successfully learn these languages, the number of learning examples needed does not depend on the size of alphabet. But it depends on the size of value of k. However, the issue of identifying the value of k for each language is a challenge research topic in application to actual languages.

6. CONCLUSION AND FUTURE WORK

In this paper we prove that there exists a characteristic set for k-acceptable language. We also demonstrate that the size of the characteristic set does not depend on the size of the alphabet, but depends on the value of k. For future work, it remains to study on learnability of this class from only positive examples.

References

[1] P. Cruz, and E. Vidal, "Two grammatical inference applications in music processing," *Applied Artificial Intelligence*, vol. 22(1-2), pp. 53-76, 2008.

[2] C. de la Higuera, "A bibliographical study of grammatical inference," *Pattern Recognition*, vol. 38, pp. 1332-1348, 2005.

[3] E. M. Gold, "Language identification in the limit," *Information and Control*, vol. 10(5), pp. 447-474, 1967.

[4] E. M. Gold, "Complexity of automaton identification from given data," *Information and Control*, vol. 37, pp. 302-320, 1978.

[5] C. de la Higuera, "Characteristic sets for polynomial grammatical inference," *Machine Learning*, vol. 27(2), pp.125-138, 1997.

[6] J. Oncina and P. Garcia, "Identifying regular languages in polynomial time," *Advances in Structural and Syntactic Pattern Recognition*, vol. 5, pp. 99-102, 1992.

[7] A.C. Gomez and G.I. Alvarez, "Learning commutative regular languages," *Lecture Notes in Artificial Intelligence*, vol. 5278, pp 71-83, 2008.

[8] S. Verwer, M.D. Weerdt and C. Witteveen, "One-clock deterministic timed automata are efficiently identification in the limit," *Lecture Notes in Computer Science*, vol. 5457, pp. 740-751, 2009.

[9] S. Verwer, M.D. Weerdt and C. Witteveen, "One-clock deterministic timed automata are efficiently identification in the limit," *Lecture Notes in Computer Science*, vol. 5457, pp. 740-751, 2009.

[10] T. Yokomori, "On polynomial-time learnability in the limit of strictly deterministic automata," *Machine Learning*, vol. 19(2), pp. 153 - 179, 1995.

[11] C. de la Higuera, "Ten open problems in grammatical inference," *Grammatical Inference Proceeding of ICGI 2006*, pp. 32-44, 2006.

[12] A. Jitpattanakul and A. Surarerks, "An algorithm for learning k-DFA from informant," *Proceeding of the 13th International Annual Symposium on Computational Science and Engineering*, pp. 31-36, 2009.

Anuchit Jitpattanakul is a Ph.D. candidate at department of computer engineering, faculty of engineering, Chaulalongkorn University. He received the B.S. degree in applied mathematics from King Mongkut's Institute of Technology North Bangkok, Bangkok, in 2000. He received the M.S. degree in computational science from Chulalongkorn University, Bangkok, in 2004. His main research interests include grammatical inference, pattern recognition, and machine learning.

Athasit Surarerks received his Ph.D. degree in Informatics from Université Pierre et Marie Curie, Paris, France in 2001. He is now an Assistant Professor in department of computer engineering, faculty of engineering, Chulalongkorn University. His current research interests include Computational Theory Algorithm and Computer Arithmetic.