数理解析研究所講究録 1205 巻 2001 年 142-147



# A Catalog for Prediction-Preserving Reducibility with Membership Queries on Formal Languages

平田耕一
Kouichi Hirata
九州工業大学情報工学部
Dept. of Artif. Intel., Kyushu Inst. of Tech.

坂本比呂志 有村博紀
Hiroshi Sakamoto Hiroki Arimura
九州大学システム情報科学研究院
Dept. of Informatics, Kyushu Univ.

**Abstract**

In this paper, we present *prediction-preserving reducibility with membership queries* on formal languages, in particular, simple CFGs and finite unions of regular pattern languages.

## 1 Introduction

The task of predicting the classification of a new example is frequently discussed from the viewpoints of both *passive* and *active* settings. In a passive setting, the examples are all chosen independently according to a fixed but unknown probability distribution, and the learner has no control over selection of examples [7, 9]. In an active setting, on the other hand, the learner is allowed to ask about particular examples, that is, the learner makes *membership queries*, before the new example to predict is given to the learner [1, 4].

Pitt and Warmuth [9] have been formalized the model of prediction and a reduction between two prediction problems that preserves polynomial-time predictability called a *prediction-preserving reduction* in a passive setting. Angluin and Kharitonov [4] have extended to the model and the reduction in an active setting. The reduction is called a *prediction-preserving reduction with membership queries* or *pwm-reduction* for short.

Concerned with language learning, we can design a polynomial-time algorithm to predict deterministic finite automata (DFAs) in an active setting [1], while predicting DFAs is as hard as computing certain apparently hard cryptographic predicates in a passive setting [7]. Furthermore, predicting nondeterministic finite automaton (NFAs) and unrestricted context-free grammars (CFGs) is also hard under the same cryptographic assumptions in an active setting [4]. Here, the *cryptographic assumptions* denote the intractability of inverting RSA encryption, recognizing quadratic residues and factoring Blum integers.

In this paper, we present the prediction-preserving reducibility with membership queries on formal languages. First, we deal with the following simple CFGs: *linear grammars* ($\mathcal{L}_{\text{linear}}$), *right-linear grammars* ($\mathcal{L}_{\text{right-linear}}$), and *left-linear grammars* ($\mathcal{L}_{\text{left-linear}}$), *k-bounded* CFGs [2] ($\mathcal{L}_{k\text{-bounded-CFG}}$), the *sequential CFGs* [5] ($\mathcal{L}_{\text{sqCFG}}$), the *properly sequential CFGs* ($\mathcal{L}_{\text{psqCFG}}$), and the *k-CFGs* ($\mathcal{L}_{k\text{-CFG}}$). Next, we introduce a *regular pattern* [10] that is a string of variables and constants of which each variable occurs at most once. A *regular pattern language* is a language consisting of strings as instances of a given pattern. Then, we deal with the *bounded finite union* of regular pattern languages by some constant $m$ ($\mathcal{L}_{\cup_m \text{RP}}$) and the *unbounded finite union* of regular pattern languages ($\mathcal{L}_{\cup\text{RP}}$) [11].

By using pwm-reduction, we present the following results: $\mathcal{L}_{\text{NFA}} \cong_{\text{pwm}} \mathcal{L}_{\text{right-linear}}$, $\mathcal{L}_{\text{NFA}} \cong_{\text{pwm}} \mathcal{L}_{\text{left-linear}}$, $\mathcal{L}_{\text{NFA}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{\text{linear}}$, $\mathcal{L}_{\text{NFA}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{k\text{-bounded-CFG}}$ for each $k \geq 1$, $\mathcal{L}_{\text{DNF}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{\text{psqCFG}}$,

$\mathcal{L}_{\rm DNF} \trianglelefteq_{\rm pwm} \mathcal{L}_{k\text{-CFG}}$ for each $k \geq 1$, $\mathcal{L}_{\rm DNF} \trianglelefteq_{\rm pwm} \mathcal{L}_{\rm sqCFG}$, $\mathcal{L}_{\cup_m \rm RP} \trianglelefteq_{\rm pwm} \mathcal{L}_{\rm DFA}$ for each $m \geq 0$, and $\mathcal{L}_{\rm DNF} \trianglelefteq_{\rm pwm} \mathcal{L}_{\cup \rm RP}$. Hence, we obtain the following predictability with membership queries.

1. $\mathcal{L}_{\rm linear}$, $\mathcal{L}_{\rm right\text{-}linear}$, $\mathcal{L}_{\rm left\text{-}linear}$ and $\mathcal{L}_{k\text{-bounded-CFG}}$ ($k \geq 1$) are not polynomial-time predictable with membership queries under the cryptographic assumptions.

2. If $\mathcal{L}_{\rm sqCFG}$, $\mathcal{L}_{\rm psqCFG}$, $\mathcal{L}_{k\text{-CFG}}$ and $\mathcal{L}_{\cup \rm RP}$ are polynomial-time predictable with membership queries, then so are DNF formulas.

3. $\mathcal{L}_{\cup_m \rm RP}$ ($m \geq 0$) is polynomial-time predictable with membership queries.

# 2 Preliminaries

## 2.1 Simple CFGs and finite unions of regular pattern languages

Let $\Sigma$ and $N$ be two non-empty finite sets of symbols such that $\Sigma \cap N = \emptyset$. A *production* $A \to \alpha$ *on* $\Sigma$ *and* $N$ is an association from a nonterminal $A \in N$ to a string $\alpha \in (N \cup \Sigma)^*$. A *context-free grammar* (*CFG*, for short) is a 4-tuple $(N, \Sigma, P, S)$, where $S \in N$ is the distinguished *start symbol* and $P$ is a finite set of productions on $\Sigma$ and $N$. Symbols in $N$ are said to be *nonterminals*, while symbols in $\Sigma$ *terminals*. Then:

- A *linear grammar* is a CFG $G = (N, \Sigma, P, S)$ such that each production in $P$ is of the forms $T \to wUv$ or $T \to w$ for $T, U \in N$ and $w, v \in \Sigma^*$. In particular, a *right-linear* (*resp.*, *left-linear*) *grammar* if it is a linear grammar such that each production is of the forms either $T \to wU$ (*resp.*, $T \to Uw$) or $T \to w$ for $T, U \in N$ and $w \in \Sigma^*$.

- A CFG $G = (N, \Sigma, P, S)$ is called *k-bounded* [2] if the right-hand side of each production in $P$ has at most $k$ nonterminals.

- A CFG $G = (N, \Sigma, P, S)$ is called *sequential* [5] if the nonterminals in $N$ are labeled $S = T_1, \cdots, T_n$ such that, for each production $T_i \to w$, $w \in (\Sigma \cup \{T_j \mid i \leq j \leq n\})^*$. In particular, A sequential CFG satisfying that, for each production $T_i \to w$, $w \in (\Sigma \cup \{T_j \mid i < j \leq n\})^*$ is called *properly sequential*.

- A CFG $G = (N, \Sigma, P, S)$ is called *a k-CFG* if $|N| \leq k$.

Let $G$ be a CFG $(N, \Sigma, S, P)$ and $\alpha$ and $\beta$ be strings in $(\Sigma \cup N)^*$. We denote $\alpha \Rightarrow_G \beta$ if there exist $\alpha_1, \alpha_2 \in (\Sigma \cup N)^*$ such that $\alpha = \alpha_1 X \alpha_2$, $\beta = \alpha_1 \gamma \alpha_2$ and $X \to \gamma \in P$. We extend the relation $\Rightarrow_G$ to the reflexive and transitive closure $\Rightarrow_G^*$. For a nonterminal $A \in N$, the *language* $L_G(A)$ *of* $A$ is the set $\{w \in \Sigma^* \mid A \Rightarrow_G^* w\}$. The *language* $L(G)$ *of* $G$ just refers to $L_G(S)$.

Let $X$ be a countable set of *variables* such that $\Sigma \cap X = \emptyset$. A *pattern* is an element of $(\Sigma \cup X)^+$. A pattern $\pi$ is called *regular* [10] if each variable in $\pi$ occurs at most once. A *substitution* is a homomorphism from patterns to patterns that maps each symbol $a \in \Sigma$ to itself. A substitution that maps some variables to empty string $\varepsilon$ is called an $\varepsilon$-*substitution*. In this paper, we do not deal with $\varepsilon$-substitution. By $\pi\theta$, we denote the image of a pattern by a substitution $\theta$. For a pattern $\pi$, the *pattern language* $L(\pi)$ is the set $\{w \in \Sigma^+ \mid w = \pi\theta$ for some substitution $\theta\}$.

## 2.2 Prediction-preserving reduction with membership queries

Let $U$ denote $\Sigma^*$. If $w$ is a string, $|w|$ denotes its length. For each $n > 0$, $U^{[n]} = \{w \in U \mid |w| \leq n\}$. A *representation of concepts* $\mathcal{L}$ is any subset of $U \times U$. We interpret an element $\langle u, w \rangle$ of $U \times U$ as consisting a *concept representation* $u$ and an *example* $w$. The example $w$

is a member of a concept $u$ if $\langle u, w\rangle \in \mathcal{L}$. Furthermore, define the *concept represented by* $u$ as $\kappa_{\mathcal{L}}(u) = \{w \mid \langle u, w\rangle \in \mathcal{L}\}$. The *set of concepts represented by* $\mathcal{L}$ is $\{\kappa_{\mathcal{L}}(u) \mid u \in U\}$.

To represent CFGs, we define the class $\mathcal{L}_{\text{CFG}}$ as the set of pairs $\langle u, w\rangle$ such that $u$ encodes a CFG $G$ and $w \in L(G)$. Also we define the classes $\mathcal{L}_{\text{linear}}$, $\mathcal{L}_{\text{right-linear}}$, $\mathcal{L}_{\text{left-linear}}$, $\mathcal{L}_{k\text{-bounded-CFG}}$, $\mathcal{L}_{\text{seqCFG}}$, $\mathcal{L}_{\text{psqCFG}}$, and $\mathcal{L}_{k\text{-CFG}}$, corresponding to a linear grammar, right-linear grammar, left-linear grammar, $k$-bounded CFG, sequential CFG, properly sequential CFG, and $k$-CFG, respectively, as similar.

To represents regular pattern languages, the class $\mathcal{L}_{\text{RP}}$ denotes the set of pairs $\langle u, w\rangle$ such that $u$ encodes a regular pattern $\pi$ and $w$ is in the concept represented by $c$ iff $w \in L(\pi)$. Furthermore, the class $\mathcal{L}_{\cup_m\text{RP}}$ of a *bounded* finite union of regular pattern languages [11] denotes the set of pairs $\langle u, w\rangle$ such that $u$ encodes $m$ and a finite set $\pi_1, \cdots, \pi_m$ of $m$ regular patterns and $w$ is in the concept represented by $c$ iff $w \in L(\pi_i)$ for at least one $\pi_i$. Similarly, the class $\mathcal{L}_{\cup\text{RP}}$ of an *unbounded* finite union of regular pattern languages [11] denotes the set of pairs $\langle u, w\rangle$ such that $u$ encodes a finite set $\pi_1, \cdots, \pi_r$ of regular patterns and $w$ is in the concept represented by $c$ iff $w \in L(\pi_i)$ for at least one $\pi_i$.

Additionally, we introduce the following classes. The class $\mathcal{L}_{\text{DFA}}$ (*resp.*, $\mathcal{L}_{\text{NFA}}$) denotes the set of pairs $\langle u, w\rangle$ such that $u$ encodes a DFA (*resp.*, NFA) $M$ and $M$ accepts $w$. The class $\mathcal{L}_{\text{DNF}}$ denotes the set of pairs $\langle u, w\rangle$ such that $u$ encodes a positive integer $n$ and a DNF formula $d$ over $n$ Boolean variables $x_1, \cdots, x_n$ such that $|w| = n$ ($w = w_1 \cdots w_n$) and the assignment $x_i = w_i$ ($1 \le i \le n$) satisfies $d$.

In order to obtain the results of this paper, it is sufficient to introduce the following concept of *prediction-preserving reducibility* [4, 9]. Hence, we omit the formal definitions of the prediction algorithm and the predictability. See the papers [4, 7, 9] for more detail.

Angluin and Kharitonov [4] have extended the prediction-preserving reduction by Pitt and Warmuth [9] with membership queries. It also a tool for showing hardness results of predicting some classes of representations with membership queries.

**Definition 1 (Angluin & Kharitonov [4])** Let $\mathcal{L}_i$ be a representation of concepts over domain $U_i$ ($i = 1, 2$). We say that *predicting* $\mathcal{L}_1$ *reduces to predicting* $\mathcal{L}_2$ *with membership queries* (*pwm-reduces*, for short), denoted by $L_1 \trianglelefteq_{\text{pwm}} \mathcal{L}_2$, if there exist an *instance mapping* $f : \mathbf{N} \times \mathbf{N} \times U_1 \to U_2$, a *concept mapping* $g : \mathbf{N} \times \mathbf{N} \times \mathcal{L}_1 \to \mathcal{L}_2$, and a *query mapping* $h : \mathbf{N} \times \mathbf{N} \times U_2 \to U_1 \cup \{\top, \bot\}$ satisfying the following conditions.

1. For each $x \in U_1^{[n]}$ and $u \in \mathcal{L}_1^{[s]}$, $x \in \kappa_{\mathcal{L}_1}(u)$ iff $f(n, s, x) \in \kappa_{\mathcal{L}_2}(g(n, s, u))$.
2. $f$ is computable in time bounded by a polynomial in $n$, $s$ and $|x|$.
3. The size of $g(n, s, u)$ is bounded by a polynomial in $n$, $s$ and $|u|$.
4. For each $x' \in U_2$ and $u \in \mathcal{L}_1^{[s]}$, if $h(n, s, x') = \top$ then $x' \in \kappa_{\mathcal{L}_2}(g(n, s, u))$; if $h(n, s, x') = \bot$ then $x' \notin \kappa_{\mathcal{L}_2}(g(n, s, u))$; if $h(n, s, x') = x \in U_1$, then it holds that $x' \in \kappa_{\mathcal{L}_2}(g(n, s, u))$ iff $x \in \kappa_{\mathcal{L}_1}(u)$.
5. $h$ is computable in time bounded by a polynomial in $n$, $s$ and $|x'|$.

If $\mathcal{L}_1 \trianglelefteq_{\text{pwm}} \mathcal{L}_2$ and $\mathcal{L}_2 \trianglelefteq_{\text{pwm}} \mathcal{L}_1$, we denote $\mathcal{L}_1 \cong_{\text{pwm}} \mathcal{L}_2$.

The following theorem is useful for showing the predictability or the hardness of predictability of the class of representations.

**Theorem 1 (Angluin & Kharitonov [4])** *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be representations of concepts and suppose that $\mathcal{L}_1 \trianglelefteq_{\text{pwm}} \mathcal{L}_2$. If $\mathcal{L}_2$ is polynomial-time predictable with membership queries, then so is $\mathcal{L}_1$. If $\mathcal{L}_1$ is not polynomial-time predictable with membership queries, then neither is $\mathcal{L}_2$.*

It is well known the following statements:

1. $\mathcal{L}_{\text{DFA}}$ *is polynomial-time predictable with membership queries* [1].

2. $\mathcal{L}_{\text{NFA}}$ *and* $\mathcal{L}_{\text{CFG}}$ *are not polynomial-time predictable with membership queries under the cryptographic assumptions* [4].

3. $\mathcal{L}_{\text{DNF}}$ *is either polynomial-time predictable or not polynomial-time predictable with membership queries, if there exist one-way functions that cannot be inverted by polynomial-sized circuits* [4].

# 3 PWM-Reducibility

In this section, we fix $f$, $g$ and $h$ to an instance mapping, a concept mapping, and a query mapping. Furthermore, the parameters $n$ and $s$ denote the bounds of examples and representations, respectively.

## 3.1 Simple CFGs

First note that, by using the equivalent transformation between a NFA and a right-linear grammar [6] as a concept mapping, we observe that $\mathcal{L}_{\text{NFA}} \cong_{\text{pwm}} \mathcal{L}_{\text{right-linear}}$. Furthermore, for a CFG $G = (N, \Sigma, P, S)$, let $G^R$ be a CFG $(N, \Sigma, P', S)$ such that $T \to w^R \in P'$ for each $T \to w \in P$. Here, $R$ denotes the reversal of a word. For a right-linear (*resp.*, left-linear) grammar $G$, construct $f$, $g$ and $h$ as $f(n,s,e) = e^R$, $g(n,s,G) = G^R$ and $h(n,s,e') = e'^R$. Then, it is obvious that $\mathcal{L}_{\text{right-linear}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{\text{left-linear}}$ (*resp.*, $\mathcal{L}_{\text{left-linear}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{\text{right-linear}}$), so it holds that $\mathcal{L}_{\text{right-linear}} \cong_{\text{pwm}} \mathcal{L}_{\text{left-linear}}$. Summary:

**Theorem 2** $\mathcal{L}_{\text{NFA}} \cong_{\text{pwm}} \mathcal{L}$ *for* $\mathcal{L} \in \{\mathcal{L}_{\text{right-linear}}, \mathcal{L}_{\text{left-linear}}\}$. *Also,* $\mathcal{L}_{\text{NFA}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{\text{linear}}$ *and* $\mathcal{L}_{\text{NFA}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{k\text{-bounded-CFG}}$ *for each* $k \geq 1$.

**Theorem 3** $\mathcal{L}_{\text{DNF}} \trianglelefteq_{\text{pwm}} \mathcal{L}$ *for* $\mathcal{L} \in \{\mathcal{L}_{\text{psqCFG}}, \mathcal{L}_{\text{sqCFG}}\}$.

*Proof.* Let $d$ be a DNF formula $t_1 \vee \cdots \vee t_m$ over $n$ Boolean variables $x_1, \ldots, x_n$. First, we define $w_i^j$ $(1 \leq i \leq n, 1 \leq j \leq m)$ as $w_i^j = 1$ if $t_j$ contains $x_i$; $w_i^j = 0$ if $t_j$ contains $\overline{x_i}$; $w_i^j = T$ otherwise. Then, construct $f$, $g$ and $h$ as follows:

$$\begin{aligned} f(n,s,e) &= e, \\ g(n,s,d) &= (\{S,T\}, \{0,1\}, S, \{S \to w_1^1 \cdots w_n^1 \mid \ldots \mid w_1^m \cdots w_n^m,\ T \to 0 \mid 1\}), \\ h(n,s,e') &= e'. \end{aligned}$$

It is obvious that the above $f$, $g$ and $h$ satisfy the conditions of Definition 1. □

**Theorem 4** *For each* $k \geq 1$, $\mathcal{L}_{\text{DNF}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{k\text{-CFG}}$.

*Proof.* Theorem 3 implies that $\mathcal{L}_{\text{DNF}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{k\text{-CFG}}$ for each $k \geq 2$. Then, it is sufficient to show that $\mathcal{L}_{\text{DNF}} \trianglelefteq_{\text{pwm}} \mathcal{L}_{1\text{-CFG}}$. Let $d = t_1 \vee \cdots \vee t_m$ be a DNF formula over $n$ Boolean variables $x_1, \ldots, x_n$. Then, define $w_i^j$ $(1 \leq i \leq n, 1 \leq j \leq m)$ as $w_i^j = 1$ if $t_j$ contains $x_i$; $w_i^j = 0$ if $t_j$ contains $\overline{x_i}$; $w_i^j = S$ otherwise. Then, construct $f$, $g$ and $h$ as follows:

$$\begin{aligned} f(n,s,e) &= e, \\ g(n,s,d) &= (\{S\}, \{0,1\}, S, \{S \to 0 \mid 1 \mid w_1^1 \cdots w_n^1 \mid \ldots \mid w_1^m \cdots w_n^m \mid \underbrace{S \cdots S}_{n+1} \mid \ldots \mid \underbrace{S \cdots S}_{2n}\}), \\ h(n,s,e') &= \begin{cases} e' & \text{if } |e'| = n, \\ \bot & \text{if } 1 < |e'| < n, \\ \top & \text{if } |e'| = 1 \text{ or } |e'| > n. \end{cases} \end{aligned}$$

For each $e \in \{0,1\}^n$, it holds that $e$ satisfies $d$ iff $S \Rightarrow^*_{g(n,s,d)} f(n,s,e)$. Furthermore, for each $e' \in \{0,1\}^*$, if $h(n,s,e') = \bot$, then $S \not\Rightarrow^*_{g(n,s,d)} e'$, because $g(n,s,d)$ generates no strings of length more than 1 and less than $n$; If $h(n,s,e') = e'$, then it holds that $S \Rightarrow^*_{g(n,s,d)} e'$ iff $h(n,s,e')$ satisfies $d$.

Finally, consider the case that $h(n,s,e') = \top$. It is sufficient to show that, for each $k \geq 1$, it holds that $S \Rightarrow^*_{g(n,s,d)} \underbrace{S\cdots S}_{kn+m}$ for each $m$ $(1 \leq m \leq n-1)$. If $k=1$, then, by the definition, it holds that $S \Rightarrow^*_{g(n,s,d)} \underbrace{S\cdots S}_{n+m}$ for each $m$ $(1 \leq m \leq n-1)$. Suppose that it holds that, for some $k \geq 1$, $S \Rightarrow^*_{g(n,s,d)} \underbrace{S\cdots S}_{kn+m}$ for each $m$ $(1 \leq m \leq n-1)$. Then, it holds that $S \Rightarrow^*_{g(n,s,d)} \underbrace{S\cdots S}_{kn+(m-1)} S \Rightarrow_{g(n,s,d)} \underbrace{S\cdots S}_{kn+(m-1)} \underbrace{S\cdots S}_{n+1} = \underbrace{S\cdots S}_{(k+1)n+m}$ for each $m$ $(1 \leq m \leq n-1)$. Hence, $g(n,s,d)$ generates all strings of length more than $n$, so if $h(n,s,e') = \top$, then $S \Rightarrow^*_{g(n,s,d)} e'$. □

## 3.2 Finite union of regular pattern languages

Since each regular pattern language is regular [10], we can construct a DFA $M_\pi$ such that $L(M_\pi) = L(\pi)$ for each regular pattern $\pi$ as follows: Suppose that $\pi$ is a regular pattern of the form $\pi = x_0\alpha_1x_1\alpha_2\cdots x_{n-1}\alpha_nx_n$, where $x_i \in X$ and $\alpha_i = a^i_1a^i_2\cdots a^i_{m_i} \in \Sigma^+$. Then, the *corresponding DFA $M_\pi$ of $\pi$* is a DFA $(\Sigma, Q, \delta, q_0, F)$ such that:

1. $Q = \{q_0, p^1_1, \ldots, p^1_{m_1}, q_1, p^2_1, \ldots, p^2_{m_2}, q_2, \ldots, q_{n-1}, p^n_1, \ldots, p^n_{m_n}, q_n\}$ and $F = \{q_n\}$,
2. $\delta(q_i, a) = p^{i+1}_1$ and $\delta(q_n, a) = q_n$ for each $a \in \Sigma$ and $0 \leq i \leq n-1$,
3. $\delta(p^i_j, a^i_j) = p^i_{j+1}$ and $\delta(p^i_{m_i}, a^i_{m_i}) = q_i$ for each $1 \leq i \leq n$ and $1 \leq j \leq m_i - 1$,
4. $\delta(p^i_j, a) = p^i_1$ for each $a \in \Sigma$ such that $a \neq a^i_j$.

It is obvious that $|M_\pi|$ is bounded by a polynomial in $|\pi|$. We can easily shown that $\mathcal{L}_{\rm RP} \trianglelefteq_{\rm pwm} \mathcal{L}_{\rm DFA}$ by constructing $f$, $g$ and $h$ for each regular pattern $\pi$ as $f(n,s,e) = e$, $g(n,s,\pi) = M_\pi$ and $h(n,s,e') = e'$. Then, $\mathcal{L}_{\rm RP}$ is polynomial-time predictable with membership queries [8].

**Theorem 5** *For each $m \geq 0$, $\mathcal{L}_{\cup_m \rm RP} \trianglelefteq_{\rm pwm} \mathcal{L}_{\rm DFA}$.*

*Proof.* Let $\pi_1, \ldots, \pi_m$ be $m$ regular patterns. Also let $M_{\pi_i} = (Q_i, \Sigma, \delta_i, q^i_0, F_i)$ be the corresponding DFA of $\pi_i$. First, construct a DFA $M_{\pi_1,\ldots,\pi_m} = (Q_1 \times\cdots\times Q_m, \Sigma, \delta, (q^1_0, \ldots, q^m_0), F_1 \times\cdots\times F_m)$ such that $\delta((q_1,\ldots,q_m),a) = (p_1,\ldots,p_m)$ iff $\delta_i(q_i,a) = p_i$ for each $i$ $(1 \leq i \leq m)$. Then, construct $f$, $g$ and $h$ as $f(n,s,e) = e$, $g(n,s,\{\pi_1,\ldots,\pi_m\}) = M_{\pi_1,\ldots,\pi_m}$ and $h(n,s,e') = e'$. Note that the size of $g(n,s,\{\pi_1,\ldots,\pi_m\})$ is bounded by a polynomial in $s$, i.e., $O(s^m)$. It is obvious that $L(\pi_1) \cup \cdots \cup L(\pi_m) = L(M_{\pi_1,\ldots,\pi_m})$, which implies that $\mathcal{L}_{\cup_m \rm RP} \trianglelefteq_{\rm pwm} \mathcal{L}_{\rm DFA}$. □

**Theorem 6** $\mathcal{L}_{\rm DNF} \trianglelefteq_{\rm pwm} \mathcal{L}_{\cup \rm RP}$.

*Proof.* Let $d = t_1 \vee \cdots \vee t_m$ be a DNF formula over $n$ Boolean variables $x_1, \ldots, x_n$. First, for each term $t_j$ $(1 \leq j \leq m)$, construct a regular pattern $\pi_j = \pi^j_1 \cdots \pi^j_n$ as $\pi^j_i = 1$ if $t_j$ contains $x_i$; $\pi^j_i = 0$ if $t_j$ contains $\overline{x_i}$; $\pi^j_i = x^j_i$ otherwise. Furthermore, let $\pi$ be a regular pattern $x_1 \cdots x_nx_{n+1}$. Then, construct $f$, $g$ and $h$ as follows:

$$\begin{aligned} f(n,s,e) &= e, \\ g(n,s,d) &= \{\pi_1, \ldots, \pi_m, \pi\}, \\ h(n,s,e') &= \begin{cases} e' & \text{if } |e'| = n, \\ \top & \text{if } |e'| > n, \\ \bot & \text{if } |e'| < n. \end{cases} \end{aligned}$$

For each $e' \in \{0,1\}^*$, we can check the properties of $h$ in Definition 1 as follows. Since $L(\pi) = \{w \in \{0,1\}^* \mid |w| \geq n+1\}$, if $h(n,s,e') = \top$, then $e' \in \kappa_{\mathcal{L}_{\cup\mathrm{RP}}}(g(n,s,d))$. On the other hand, since $|\pi_j| = n$ $(1 \leq j \leq m)$ and $|\pi| = n+1$, $\kappa_{\mathcal{L}_{\cup\mathrm{RP}}}(g(n,s,d))$ contains no strings of length $< n$. So, if $h(n,s,e') = \perp$, then $e' \notin \kappa_{\mathcal{L}_{\cup\mathrm{RP}}}(g(n,s,d))$. Otherwise, i.e, if $h(n,s,e') = e'$, note that $|e'| = n$, so $e' \notin L(\pi)$. Then, $e' \in L(\pi_1) \cup \cdots \cup L(\pi_m)$. Thus, there exists an index $i$ $(1 \leq i \leq m)$ such that $e' \in L(\pi_i)$ iff $e'$ is obtained by replacing the variables in $\pi_i$ with 0 or 1, which is corresponding to a truth assignment satisfying $t_i$. Hence, $e' \in \kappa_{\mathcal{L}_{\cup\mathrm{RP}}}(g(n,s,d))$ iff $e' \in \kappa_{\mathcal{L}_{\mathrm{DNF}}}(d)$.

Furthermore, for each $e \in \{0,1\}^n$, $e \in \kappa_{\mathcal{L}_{\mathrm{DNF}}}(d)$ iff $f(n,s,e) \in \kappa_{\mathcal{L}_{\cup\mathrm{RP}}}(g(n,s,d))$. Hence, it holds that $\mathcal{L}_{\mathrm{DNF}} \trianglelefteq_{\mathrm{pwm}} \mathcal{L}_{\cup\mathrm{RP}}$. □

Shinohara and Arimura [11] have discussed the inferability of $\mathcal{L}_{\cup_m\mathrm{RP}}$ and $\mathcal{L}_{\cup\mathrm{RP}}$ in the framework of inductive inference. They have shown that $\mathcal{L}_{\cup_m\mathrm{RP}}$ is inferable from positive data, whereas $\mathcal{L}_{\cup\mathrm{RP}}$ is not. In contrast, by Theorem 5 and 6, $\mathcal{L}_{\cup_m\mathrm{RP}}$ is polynomial-time predictable with membership queries, whereas $\mathcal{L}_{\cup\mathrm{RP}}$ is not polynomial-time predictable with membership queries if neither are DNF formulas.

## References


[1] D. Angluin, *Learning regular sets from queries and counterexamples*, Inform. Comput. **75** (1987) 87–106.

[2] D. Angluin, *Learning k-bounded context-free grammars*, Technical Report YALEU/DCS/RR-557, Yale University, 1987.

[3] D. Angluin, *Queries and concept learning*, Mach. Learn. **2** (1988) 319–342.

[4] D. Angluin, M. Kharitonov, *When won't membership queries help?*, J. Comput. System Sci. **50** (1995) 336–355.

[5] A. Ginsburg, *The mathematical theory of context free languages* (McGraw-Hill, 1966).

[6] J. E. Hopcroft, J. D. Ullman, *Introduction to automata theory, languages and computation* (Addison-Wesley, 1979).

[7] M. Kearns, L. Valiant, *Cryptographic limitations on learning Boolean formulae and finite automata*, J. ACM **41** (1994) 67–95.

[8] S. Matsumoto, A. Shinohara, *Learning pattern languages using queries*, in: Proc. 3rd Euro. Conf. on Computational Learning Theory, LNAI **1208** (Springer, 1997) 185–197.

[9] L. Pitt, M. K. Warmuth, *Prediction-preserving reduction*, J. Comput. System Sci. **41** (1990) 430–467.

[10] T. Shinohara, *Polynomial time inference of extended regular pattern languages*, in: Proc. RIMS Symposia on Software Science and Engineering, LNCS **147** (Springer, 1982) 115–127.

[11] T. Shinohara, H. Arimura, *Inductive inference of unbounded unions of pattern languages from positive data*, Theor. Comput. Sci. **241** (2000) 191–209.