Vol.2010-MPS-77 No.4
2010/3/4

線形分離オートマトンの最小化アルゴリズム

沼 井 裕 二 [†1]　宇田川　祥彰 [†2]　小　林　聡[†1]

本論文では，線形分離オートマトン（LSA）の最小化アルゴリズムを提示する．LSA は有限オートマトンを拡張したモデルで，実ベクトル系列を受理する能力を持ち，各状態には重み関数と閾値系列が付随する．この二つによって，各時点でのある状態からの遷移先状態が決定する．

以前の我々の論文では， LSA と， 状態数最小の LSA との特徴付けのみを行った．与えられた LSA M の状態数の最小化は，本論文のアルゴリズムによって可能となる．アルゴリズムの時間計算量は $O((K+k)\,n^2)$ である．K は M の各重みに割り当てられた閾値系列の値の数の最大値であり，k は M のある状態から出て行く辺の最大の本数であり，n は M の状態数である．

最後に， LSA の各状態における閾値系列の長さの最小化について論ずる．

Minimization Algorithm of Linear Separation Automata

Yuji Numai ,[†1] Yoshiaki Udagawa [†2]
and Satoshi Kobayashi [†1]

In this paper, we present a minimization algorithm of a linear separation automaton (LSA). An LSA is an extended model of a finite automaton. It accepts a sequence of real vectors, and has a weight function and a threshold sequence at every state, which determine the transition from some state to another at each step.

In our previous paper, we characterized an LSA and the minimum state LSA. The minimum state version for a given LSA M is obtained by the algorithm presented in this paper. Its time complexity is $O((K+k)\,n^2)$, where K is the maximum number of threshold values assigned to each weight, k is the maximum number of edges going out from a state of M, and n is the number of states in M.

Moreover, we discuss on the minimization of the length of a threshold sequence at each state.

1. Introduction

A finite automaton can be extended to deal with real values in some sense. Such extensions include a hybrid automaton[1),7)] and a timed automaton[3)]. Many researchers utilize computational models that can deal with real values to solve various problems including weather forecasting[6)], motion recognition[4),5)], and time-sequential image analysis[10)]. Therefore, we believe that the establishment of the theory of automata that can deal with real values is very important.

In our previous paper[8)], we theoretically analyzed a **linear separation automaton (LSA)**. It accepts a sequence of real vectors, and has a weight function and a threshold sequence at every state, which determine the transition from some state to another at each step. We proved Myhill-Nerode theorem for LSA, established the uniqueness of the minimum state LSA for a given one, and characterized the minimum state LSA for a given one.

This paper presents an algorithm to minimize the number of states of a given LSA M. Its time complexity is $O((K+k)\,n^2)$, where K is the maximum number of threshold values assigned to each weight, k is the maximum number of edges going out from a state of M, and n is the number of states in M. We moreover discuss on the minimization of the length of a threshold sequence at each state.

2. Preliminaries

In this section we introduce basic definitions and notation needed in this paper.

By $\mathbf{R}$, we denote the set of real numbers. For a positive integer d, by $\mathbf{R}^d$ we denote a d-dimensional vector space over $\mathbf{R}$. For $x, y \in \mathbf{R}^d$, $x \otimes y$ denotes the inner product of x and y. We define $(\mathbf{R}^d)^*$ as the set of all finite sequences of vectors in $\mathbf{R}^d$.

†1 電気通信大学大学院電気通信学研究科情報工学専攻
Department of Computer Science, Graduate School of Electro-Communications, The University of Electro-Communications
†2 エヌ・ティ・ティ アイティ株式会社
NTT-IT Corporation

Vol.2010-MPS-77 No.4
2010/3/4

For a sequence $\alpha = \langle x_1, \ldots, x_n \rangle \in (\mathbf{R}^d)^*$, we denote the length of α by $|\alpha|$, that is, $|\alpha| = n$. An element in $(\mathbf{R}^d)^*$ of length 0 is called an empty sequence, and is denoted by λ. For sequences $\alpha, \beta \in (\mathbf{R}^d)^*$, we denote the concatenation of α and β by $\alpha\beta$. For $\alpha = \langle x_1, \ldots, x_n \rangle \in (\mathbf{R}^1)^*$, the sequence α is said to be **increasing** if the inequality $x_i < x_{i+1}$ holds for every i.

A partition $\pi = \{S_1, \ldots, S_k\}$ of $\mathbf{R}^d$ (i.e., $S_1, \ldots, S_k$ are mutually disjoint non-empty subsets of $\mathbf{R}^d$ such that $\cup_{i=1,\ldots,k} S_i = \mathbf{R}^d$) is said to be **linearly separable** iff there exist $w \in \mathbf{R}^d$ and an increasing $h = \langle h_1, \ldots, h_{k-1} \rangle \in (\mathbf{R}^1)^*$ such that, for any $x \in \mathbf{R}^d$,

$$h_{i-1} < w \otimes x \leq h_i \quad \Leftrightarrow \quad x \in S_i \quad (i = 1, \ldots, k)$$

holds, where $h_0 = -\infty$ and $h_k = \infty$.

Consider equivalence relations $\equiv, \equiv_1$, and $\equiv_2$ over $(\mathbf{R}^d)^*$. The number of the equivalence classes of $\equiv$ is called the **index** of $\equiv$. An equivalence relation $\equiv_1$ is **finer** than an equivalence relation $\equiv_2$ (or $\equiv_2$ is coarser than $\equiv_1$) iff $x \equiv_1 y$ implies $x \equiv_2 y$ for any x and y. An equivalence relation $\equiv$ is **right invariant** iff $\alpha \equiv \beta$ implies $\alpha\gamma \equiv \beta\gamma$ for any α, β and γ.

Consider partitions π_1 and π_2 of $\mathbf{R}^d$. A partition π_1 is **finer** than a partition π_2 (or π_2 is coarser than π_1) iff for any block $B_1 \in \pi_1$, there exists a block $B_2 \in \pi_2$ such that $B_1 \subseteq B_2$. We say that π_1 is a **refinement** of π_2 iff π_1 is finer than π_2.

Lemma 1. Let w, w' be unit vectors in $\mathbf{R}^d$ such that $w \neq w'$, and consider any $h \in \mathbf{R}$. There exists $h' \in \mathbf{R}$ such that for any $\varepsilon > 0$, there exist $x_1, x_2 \in \mathbf{R}^d$ satisfying

$$h' - \varepsilon \leq w' \otimes x_2 \leq w' \otimes x_1 = h' \quad \text{and} \quad w \otimes x_1 = h < w \otimes x_2 \leq h + \varepsilon \ .$$

□

This lemma means that $w' \otimes x_1$ and $w' \otimes x_2$ can be put together closely enough in the interval $[h' - \varepsilon, h']$, and that $w \otimes x_1$ and $w \otimes x_2$ are separated w.r.t. the threshold value h.

3. Linear Separation Automata and their Theoretical Results

In this section we introduce a linear separation automaton (LSA) and give some theoretical results, which have been proved in our previous paper[8].

3.1 Overview

An LSA is an extended model of a finite automaton. It accepts a sequence of real vectors, and has a weight function and a threshold sequence at every state. The transition from the current state to another is determined by comparing the inner product of the weight and input vectors with each element in the threshold sequence. Figure 1 is a state transition diagram of an LSA M_1.

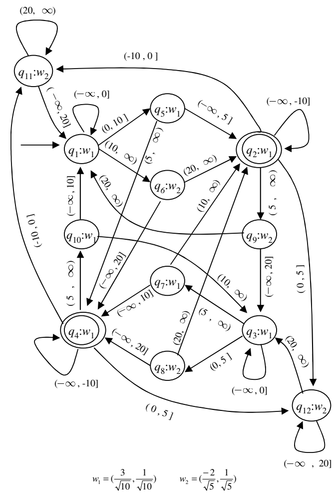


Fig. 1 LSA M_1.

Consider a state transition diagram as in Figure 1, some interval $I \subseteq \mathbf{R}$ associated with some state transition from a state p is constructed with the threshold sequence of p. Let an interval I_0 be associated with a state transition from a state q to r. If, in the current state q, the inner product of the weight and input vectors is in I_0, then the next state is r.

Example 1. Consider an LSA M_1 in Figure 1. The LSA M_1 has the weight function w, the threshold sequence h, and the state transition function δ. Let $\alpha = \langle x_1, x_2, x_3 \rangle$ be an input sequence of vectors in $\mathbf{R}^2$ with $x_1 = (3\sqrt{10}, 2\sqrt{10}), x_2 = (-\sqrt{5}, 2\sqrt{5})$, and $x_3 = (-3\sqrt{10}, -2\sqrt{10})$. The inner product $w(q_1) \otimes x_1 = 11$ is in the interval $(10, \infty)$, which implies that $\delta(q_1, x_1) = q_6$. We see in the same way that $\delta(q_6, x_2) = q_4$ and $\delta(q_4, x_3) = q_4$. The state q_4 is a final state and thus the sequence α is accepted by M_1.

□

3.2 Definitions and Notation

An LSA M is formally defined as an 8-tuple

$$M = (d, Q, q_0, F, w, h, s, \delta) \ ,$$

where

d is a positive integer specifying the dimension of input vectors to M,

Q is a finite set of states,

q_0 is an initial state ($q_0 \in Q$),

F is a finite set of final states ($F \subseteq Q$),

w is a weight function from Q to $\mathbf{R}^d$ such that $w(q)$ is a *unit vector* for any $q \in Q$,

h is a threshold function from Q to $(\mathbf{R}^1)^*$ such that $h(q)$ is increasing for every $q \in Q$, and is denoted by $h(q) = \langle h(q)_1, \ldots, h(q)_{|h(q)|} \rangle$, and

s is a sub-transition function from Q to Q^*, and is denoted by $s(q) = \langle s(q)_1, \ldots, s(q)_{|s(q)|} \rangle$.

If $|s(q)| \geq 1$, then the equality $|h(q)| = |s(q)| - 1$ holds for every $q \in Q$.

In order to improve the readability, we write $i_q = |h(q)|$ for any $q \in Q$.

δ is a state transition function from $Q \times \mathbf{R}^d$ to Q; and is defined in the following way by using w, h, and s. Consider any state $q \in Q$ and vector $x \in \mathbf{R}^d$. The definition of δ is separated into three components.

First, in the case of $|s(q)| = 0$, the value $\delta(q, x)$ is undefined.

Secondly, suppose that $|s(q)| = 1$. The value $\delta(q, x)$ is defined as $\delta(q, x) = s(q)_1$.

Finally, assume that $|s(q)| \geq 2$. The value $\delta(q, x)$ is defined as follows:

$$\delta(q, x) = \begin{cases} s(q)_1 & \text{if} \quad w(q) \otimes x \leq h(q)_1 \\ s(q)_2 & \text{if} \quad h(q)_1 < w(q) \otimes x \leq h(q)_2 \\ \vdots & \quad \vdots \\ s(q)_{i_q} & \text{if} \quad h(q)_{i_q - 1} < w(q) \otimes x \leq h(q)_{i_q} \\ s(q)_{i_q+1} & \text{if} \quad h(q)_{i_q} < w(q) \otimes x \ . \end{cases}$$

Consider a state transition diagram as in Figure 1. Suppose that $\delta(q, x) = p$ holds if $h(q)_i < w(q) \otimes x \leq h(q)_{i+1}$. In the diagram, the transition from q to p is associated with the interval $(h(q)_i, h(q)_{i+1}]$.

For $\alpha = \langle x_1, \ldots, x_l \rangle \in (\mathbf{R}^d)^*$, we write $\delta(p, \alpha) = q$ if there exists a sequence $p_1 (= p), p_2, \ldots, p_{l+1} (= q)$ of states such that $\delta(p_i, x_i) = p_{i+1}$ holds for any i. We define the set of sequences accepted by an LSA M, denoted by $L(M)$, as

$$L(M) = \{ \alpha \in (\mathbf{R}^d)^* \mid \delta(q_0, \alpha) \in F \} \ .$$

A subset L of $(\mathbf{R}^d)^*$ is said to be **regular** if there exists an LSA M such that $L = L(M)$. We define the size of M as the cardinality of Q, i.e., size$(M) = |Q|$.

A state $q \in Q$ is said to be **reachable** if there exists $\alpha \in (\mathbf{R}^d)^*$ such that $\delta(q_0, \alpha) = q$. A state $q \in Q$ is said to be **unreachable** if q is not reachable.

3.3 Theoretical Results

The theorems and lemmas in this subsection have been proved in 8).

Let $\equiv$ be a right invariant equivalence relation over $(\mathbf{R}^d)^*$, and consider an equivalence class $[\alpha]_\equiv$ containing $\alpha \in (\mathbf{R}^d)^*$. An equivalence relation $R([\alpha]_\equiv)$ over $\mathbf{R}^d$ induced by $[\alpha]_\equiv$ is defined as follows:

$$x \ R([\alpha]_\equiv) \ y \quad \stackrel{\text{def}}{\Leftrightarrow} \quad \alpha x \equiv \alpha y \ .$$

For any α and β with $\alpha \equiv \beta$, the equality $R([\alpha]_\equiv) = R([\beta]_\equiv)$ holds, because $\equiv$ is right invariant.

We say that a right invariant equivalence relation $\equiv$ over $(\mathbf{R}^d)^*$ is **right linearly separable** iff for any equivalence class $[\alpha]_\equiv$, there exists a finite linearly separable partition of $\mathbf{R}^d$ that is finer than $\mathbf{R}^d / R([\alpha]_\equiv)$.

Definition 1 (Modified Myhill-Nerode Relation for LSAs). Let $S \subseteq (\mathbf{R}^d)^*$ be a set of sequences. The equivalence relation $\equiv$ over $(\mathbf{R}^d)^*$ satisfying the following conditions is called a **modified Myhill-Nerode relation** with respect to S.

(1) The equivalence relation $\equiv$ is right invariant.

(2) The equivalence relation $\equiv$ is of finite index.

(3) The equivalence relation $\equiv$ is right linearly separable.

(4) The set S is a union of some equivalence classes of $\equiv$.

Vol.2010-MPS-77 No.4
2010/3/4

□

For any subset S of $(\mathbf{R}^d)^*$, we define an equivalence relation $\approx_S$ over $(\mathbf{R}^d)^*$ as follows:

$$\alpha \approx_S \beta \overset{\text{def}}{\Leftrightarrow} \forall \gamma \in (\mathbf{R}^d)^* \ \ (\alpha\gamma \in S \ \textbf{iff}\ \beta\gamma \in S) \ .$$

Theorem 1 (**Myhill-Nerode Theorem for LSAs).** Let $S \subseteq (\mathbf{R}^d)^*$ be a set of sequences. The following three statements are equivalent.

(1) The set S is regular.

(2) There exists a modified Myhill-Nerode relation with respect to S.

(3) The equivalent relation $\approx_S$ is of finite index and right linearly separable.

□

Theorem 1 characterizes the class of languages accepted by an LSA. Moreover, the equivalence relation $\approx_S$ is utilized to characterize the minimum state LSA.

Let $S \subseteq (\mathbf{R}^d)^*$ be a set of sequences, and α be an element in $(\mathbf{R}^d)^*$. Since $\approx_S$ is right linearly separable, there exists a finite linearly separable partition $\pi = \{S_1, \ldots, S_k\}$ which is finer than the equivalence classes of $R([\alpha]_{\approx_S})$. Thus, there exist $w_\alpha \in \mathbf{R}^d$ and $h_\alpha = \langle h_1, \ldots, h_{k-1}\rangle \in (\mathbf{R}^1)^*$ such that

$$h_{i-1} < w_\alpha \otimes x \le h_i \quad \Leftrightarrow \quad x \in S_i \ \ (i = 1, \ldots, k) \ ,$$

where $h_0 = -\infty$ and $h_k = \infty$. We define

$$M_{\min} = (d, Q_{\min}, q_{0\min}, F_{\min}, w_{\min}, h_{\min}, s_{\min}, \delta_{\min})$$

as follows:

$$Q_{\min} = (\mathbf{R}^d)^*/\approx_S \ , \qquad q_{0\min} = [\lambda]_{\approx_S} \ , \qquad F_{\min} = \{[\alpha]_{\approx_S} \mid \alpha \in S\} \ ,$$
$$\delta_{\min}([\alpha]_{\approx_S}, x) = [\alpha x]_{\approx_S} \ , \qquad w_{\min}([\alpha]_{\approx_S}) = w_\alpha \ , \qquad h_{\min}([\alpha]_{\approx_S}) = h_\alpha \ .$$

Let $M = (d, Q, q_0, F, w, h, s, \delta)$ and $M' = (d, Q', q'_0, F', w', h', s', \delta')$ be LSAs. We say that M is **isomorphic** to M' iff there exists a bijection f from Q to Q' satisfying the following conditions:

(1) $f(q_0) = q'_0$.

(2) $f(\delta(q, x)) = \delta'(f(q), x)$ holds for any $q \in Q$ and $x \in \mathbf{R}^d$.

(3) $f(F) = F'$.

Theorem 2 (**Uniqueness of Minimum State LSA).** Let S be a regular subset of $(\mathbf{R}^d)^*$. The LSA $M_{\min}$ is isomorphic to every minimum state LSA accepting S.

□

Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA accepting S with no unreachable states. For any $p, q \in Q$, there exists $\alpha, \beta \in (\mathbf{R}^d)^*$ such that $\delta(q_0, \alpha) = p$ and $\delta(q_0, \beta) = q$. We define the equivalence relation $\sim$ over Q as follows:

$$p \sim q \overset{\text{def}}{\Leftrightarrow} \alpha \approx_S \beta \ .$$

The states p and q are said to be **indistinguishable** iff $p \sim q$. The states p and q are said to be **distinguishable** iff $p \not\sim q$.

For any $p \in Q$, by $r(p)$ we denote a **representative element** of $[p]_\sim$. We define an LSA

$$M/\sim = (d, Q', q'_0, F', w', h', s', \delta') \ ,$$

where

$$Q' = Q/\sim \ , \quad q'_0 = [q_0]_\sim \ , \quad F' = \{[q]_\sim \mid q \in F\} \ , \quad \delta'([q]_\sim, x) = [\delta(r(q), x)]_\sim \ ,$$
$$w'([q]_\sim) = w(r(q)) \ , \quad h'([q]_\sim) = h(r(q)) \ .$$

Theorem 3 (**Characterization of Minimum State LSA).** Let M be an LSA. The LSA $M/\sim$ is a minimum state LSA for M such that $L(M/\sim) = L(M)$.

□

4. Minimization Algorithm

In this section, we deal with an algorithm to minimize a given LSA. This algorithm is similar to that to minimize a given finite automaton.

4.1 Coarsest Refinement Approach

Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA. For a state $q_1, q_2 \in Q$, we write $q_1 \sim_w q_2$ if $w(q_1) = w(q_2)$. A state q_1 is **preceding to** a state q_2 **with respect to** a state q, denoted by $q_1 \prec_q q_2$, if there exists an integer i such that $s(q)_i = q_1$ and $s(q)_{i+1} = q_2$. For $q \in Q$, we define

$$\delta(q) = \{\, p \mid p \in s(q)\,\} \ .$$

For a subset X of Q, we define

$$\delta(X) = \{\, p \mid q \in X, \, p \in \delta(q)\,\} \ .$$

Lemma 2. Consider $q, q' \in Q$ such that $q \not\sim_w q'$ and $|\delta(q)| > 1$. For any states

p_1, p_2 with $p_1 \prec_q p_2$, there exists $x_1, x_2 \in \mathbf{R}^d$ such that $\delta(q, x_i) = p_i$ $(i = 1, 2)$ and $\delta(q', x_1) = \delta(q', x_2)$.

□

For a partition π of Q and $q_1, q_2 \in Q$, we write $q_1 \sim_{(\pi)} q_2$ if there exists $B \in \pi$ such that $q_1, q_2 \in B$. For a subset X of Q, we define

$$W(X) = \{\, w(q) \mid q \in X \,\} \ .$$

For a subset X of Q and $\omega \in W(Q)$, we define

$$X_\omega = \{\, q \in X \mid w(q) = \omega \,\} \ .$$

For any $\omega \in W(Q)$, we also define

$$H(\omega) = \{\, h(q)_i \mid q \in Q_\omega,\ 1 \le i \le i_q,\ s(q)_i \ne s(q)_{i+1} \,\} \cup \{\infty\} \ .$$

Example 2 below helps to understand these complex definitions.

For $\omega \in W(Q)$ and $v \in H(\omega)$, we define the function $\delta_{\omega,v}$ from Q_ω to Q as follows:

$$\delta_{\omega,v}(q) = \delta(q, x) \quad \text{for some} \quad x \in \mathbf{R}^d \quad \text{with} \quad \omega \otimes x = v \ .$$

We define the set of functions $\overline{\delta}$ as follows:

$$\overline{\delta} = \{\, \delta_{\omega,v} \mid \omega \in W(Q),\ v \in H(\omega) \,\} \ .$$

In the sequel, for simple description of the algorithm, we often use graph representation of mappings $f \in \overline{\delta}$ and $\delta : Q \to 2^Q$, i.e., f is represented as a graph containing edges between q_1 and q_2 such that $q_2 = f(q_1)$, and δ is represented as a graph containing edges between q_1 and q_2 such that $q_2 \in \delta(q_1)$.

Example 2. Consider an LSA M_1 in Figure 1. We have $W(Q) = \{w_1, w_2\}$, $H(w_1) = \{-10, 0, 5, 10, \infty\}$, and $H(w_2) = \{20, \infty\}$. A part of Functions in the set $\overline{\delta}$ are represented in Figure 2.

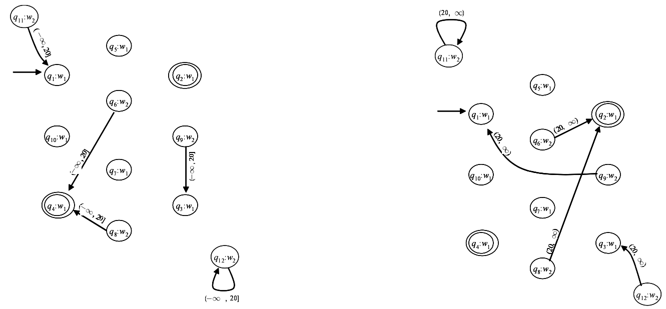
□

Theorem 4 (**Characterization of Partition** $Q/\sim$**).** Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA. The partition $Q/\sim$ is a coarsest refinement π of $\pi_0 = \{F, Q - F\}$ which satisfies the following conditions:

(C1) $\forall B \in \pi\ \ \forall f \in \overline{\delta}\ \ \exists B' \in \pi$ such that $f(B) \subseteq B'$,

(C2) $\forall B \in \pi\ \ (\ |W(B)| > 1\ \Rightarrow \exists B' \in \pi\ \ \text{such that}\ \ \delta(B) \subseteq B'\ )$.

□



$\delta_{w_2,20}$ $\delta_{w_2,\infty}$

Fig. 2 Graphs in $\overline{\delta}$ for the weight w_2 .

4.2 Minimization Algorithm

Our algorithm uses two primitive refinement operations $split_1$ and $split_2$; the former is for the condition **(C1)**, and the latter is for **(C2)**.

For a set $S \subseteq Q$, $f \in \overline{\delta}$, and a partition π of Q, the operation $split_1(S, f, \pi)$ is defined as follows:

---

find all blocks $B \in \pi$ such that $f(B) \cap S \ne \emptyset$ and $f(B) \not\subseteq S$. Define $B_1 = B \cap f^{-1}(S)$ and $B_2 = B - B_1$, and split $B \in \pi$ into the blocks B_1 and B_2, which results in the refinement of π.

For a set $S \subseteq Q$ and a partition π, $split_2(S, \pi)$ is defined as follows:

---

find all blocks $B \in \pi$ such that $\delta(B) \cap S \ne \emptyset$, $\delta(B) \not\subseteq S$ and $|W(B)| > 1$, and split B into some smaller blocks defined in the following way; Let B' be the set of states $q \in B$ such that $\delta(q) \cap S \ne \emptyset$ and $\delta(q) \not\subseteq S$. Define $B_1 = \{q \in B - B' \mid \delta(q) \subseteq S\}$ and $B_2 = (B - B') - B_1$. For each $\omega \in W(B')$, consider B'_ω. Then, split $B \in \pi$ into B_1, B_2 and B'_ω's for all $\omega \in W(B')$, which results in the refinement of π.

These operations are also illustrated in Figure 3 and Figure 4.

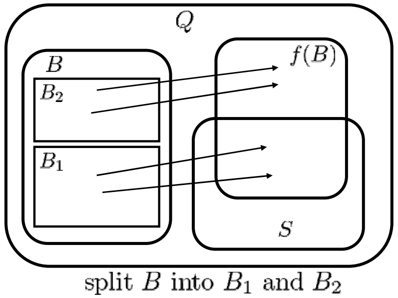


Fig. 3 $split_1(S, f, \pi)$.

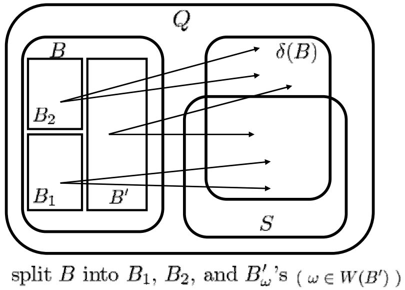


Fig. 4 $split_2(S, \pi)$.

Now, we present an algorithm to minimize a given LSA, Algorithm 1. This algorithm checks the existence of a block B with which splitting operations ($split_2$ first, and then $split_1$) can be applied to the current partition. This process is continued until no more refinement is possible.

Algorithm 1 Minimization Algorithm for LSA

Input: An LSA $M = (d, Q, q_0, F, w, h, s, \delta)$

Output: π

```
let π = {F, Q − F};
loop
    if ∃ B ∈ π such that split2(B, π) ≠ π then
        replace π with split2(B, π);
    else if ∃ B ∈ π, ∃ f ∈ δ̄ such that split1(B, f, π) ≠ π then
        replace π with split1(B, f, π);
    else
        output π and halt;
    end if
end loop
```

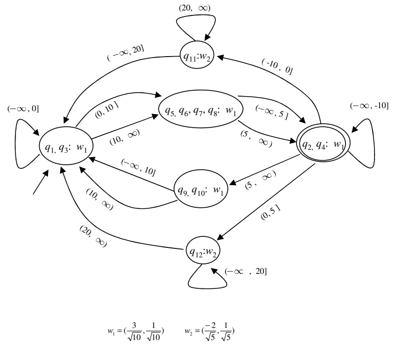


Fig. 5 Minimum state LSA of M_1.

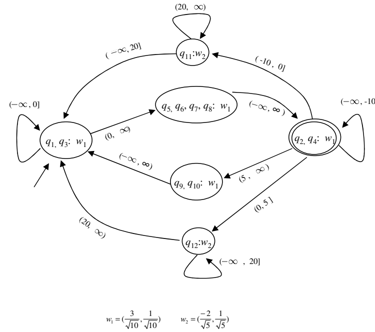


Fig. 6 Optimized version of LSA in Figure 5.

4.3 Example Run

We show an example run of Algorithm 1 for the input M_1 in Figure 1.

Let $\pi = \{B_1, B_2\}$, where

$$B_1 = \{q_2, q_4\}, B_2 = \{q_1, q_3, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}\}.$$

First, $split_2(B_1, \pi)$ constructs the new partition $\pi_1 = \{B_1, B_3, B_4\}$, where

$$B_1 = \{q_2, q_4\}, B_3 = \{q_5, q_6, q_7, q_8\}, B_4 = \{q_1, q_3, q_9, q_{10}, q_{11}, q_{12}\}.$$

Next, $split_2(B_4, \pi_1)$ constructs the new partition $\pi_2 = \{B_1, B_3, B_5, B_6\}$, where

$$B_1 = \{q_2, q_4\}, B_3 = \{q_5, q_6, q_7, q_8\}, B_5 = \{q_1, q_3\}, B_6 = \{q_9, q_{10}, q_{11}, q_{12}\}.$$

Next, $split_2(B_5, \pi_2)$ constructs the new partition $\pi_3 = \{B_1, B_3, B_5, B_7, B_8\}$, where

$$B_1 = \{q_2, q_4\}, B_3 = \{q_5, q_6, q_7, q_8\}, B_5 = \{q_1, q_3\}, B_7 = \{q_9, q_{10}\}, B_8 = \{q_{11}, q_{12}\}.$$

Finally, $split_1(B_8, \delta_{w_2,20}, \pi_3)$ constructs the new partition $\pi_4 = \{B_1, B_3, B_5, B_7, B_9, B_{10}\}$, where

$B_1 = \{q_2, q_4\}, B_3 = \{q_5, q_6, q_7, q_8\}, B_5 = \{q_1, q_3\}, B_7 = \{q_9, q_{10}\}, B_9 = \{q_{11}\}, B_{10} = \{q_{12}\}$.

No more refinement is possible. Therefore Algorithm 1 outputs π_4 and halts.

The minimum state LSA for M_1 with the set π_4 of states is in Figure 5.

4.4 Correctness of Algorithm

We give some basic properties of these operations:

Lemma 3. A partition π satisfies **(C1)** if and only if $split_1(B, f, \pi) = \pi$ for every block

Vol.2010-MPS-77 No.4
2010/3/4

$B \in \pi$ and $f \in \overline{\delta}$. A partition π satisfies **(C2)** if and only if $split_2(B, \pi) = \pi$ for every block $B \in \pi$.

□

Lemma 4. If π_2 is a refinement of π_1 and $split_1(S, f, \pi_1) = \pi_1$ holds, then $split_1(S, f, \pi_2) = \pi_2$ holds. If π_2 is a refinement of π_1 and $split_2(S, \pi_1) = \pi_1$ holds, then $split_2(S, \pi_2) = \pi_2$ holds.

□

Lemma 5. The equalities $split_1(S_1, f, \pi) = \pi$ and $split_1(S_2, f, \pi) = \pi$ imply $split_1(S_1 \cup S_2, f, \pi) = \pi$. The equalities $split_2(S_1, \pi) = \pi$ and $split_2(S_2, \pi) = \pi$ imply $split_2(S_1 \cup S_2, \pi) = \pi$.

□

Lemma 6. If π_1 is a refinement of π_2 and $split_2(S, \pi_2) = \pi_2$ holds, then $split_1(S, f, \pi_1)$ is a refinement of $split_1(S, f, \pi_2)$.

□

Lemma 7. Let π_1 be a partition satisfying **(C1)** and S be a union of some blocks in π_1. If π_1 is a refinement of π_2, then $split_2(S, \pi_1)$ is a refinement of $split_2(S, \pi_2)$.

□

Lemma 8. Algorithm 1 maintains the invariant that any coarsest refinement of the initial partition $\{F, Q - F\}$ satisfying **(C1)** and **(C2)** is also a refinement of the current partition π.

□

The following theorem shows the correctness of Algorithm 1.

Theorem 5 (Correctness of Algorithm 1). Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA, and $n = |Q|$. Algorithm 1 for the input M is correct and terminates after at most $n - 1$ refinement steps, having computed the coarsest refinement of $\{F, Q - F\}$ satisfying **(C1)** and **(C2)**.

Proof. Since the number of blocks of a partition of Q is less than or equal to n, and since the number of blocks increases at each refinement step, the algorithm terminates at most $n - 1$ refinement steps. Lemma 3 implies that the final partition π_f satisfies **(C1)** and **(C2)**. Moreover, Lemma 8 implies that π_f should be the coarsest refinement of $\{F, Q - F\}$ satisfying **(C1)** and **(C2)**. □

Let us discuss the time complexity of Algorithm 1. We define

$$K = \max\{\, |H(\omega)| \mid \omega \in W(Q) \,\}$$

and

$$k = \max\{\, |\delta(q)| \mid q \in Q \,\} \ .$$

The following theorem holds.

Theorem 6 (Time Complexity of Algorithm 1). Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA, and $n = |Q|$. The time complexity of Algorithm 1 for the input M is $O((K + k)\, n^2)$.

Proof. Let $m = (K + k)n$, i.e., m is the upper bound of the total number of edges contained in the graphs $f \in \overline{\delta}$ and in the graph δ. It is straightforward to see that finding a block B satisfying the if-conditions (at lines 3 and 5) and refining π afterwards can be done in time $O(m)$.

Moreover, the upper bound of the number of refining π is $n - 1$.

Hence the time complexity of Algorithm 1 is $O(mn) = O((K + k)\, n^2)$. □

5. Minimization of Length of Threshold Sequences

Up to now, we discussed on the minimization of the number of states for a given LSA, and not on that of the length of a threshold sequence at each state. In actuality, different minimum state LSAs for a given LSA might have different length of threshold sequences at some states.

In this subsection, we will elucidate some important properties of LSAs related to the threshold sequence, and minimize the length of a threshold sequence at each state of an LSA.

Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA. Consider $q \in Q$ and $x, y \in \mathbf{R}^d$ such that $\delta(q, x) = s(q)_i$ and $\delta(q, y) = s(q)_{i+1}$. If $s(q)_i = s(q)_{i+1}$, then the threshold value $h(q)_i$ is not necessary for the linear separation. Therefore it is better to remove such unnecessary threshold values.

For any $q \in Q$, we say that q is **optimized** iff $s(q)_{i-1} \neq s(q)_i$ holds for any i. We also say that M is **optimized** iff q is optimized for any $q \in Q$.

Lemma 9. Let $M = (d, Q, q_0, F, w, h, s, \delta)$ and $M' = (d, Q', q'_0, F', w', h', s', \delta')$ be LSAs. If M is isomorphic to M' w.r.t. the isomorphism f, then $w(q) = w'(f(q))$ holds for any $q \in Q$ such that $|\delta(q)| > 1$.

□

For any $q \in Q$, we define

$$H(q) = \{h(q)_i \mid s(q)_i \neq s(q)_{i+1}, 1 \leq i \leq i_q\} \ .$$

All the unnecessary threshold values in $h(q)$ are removed from $H(q)$.

The following theorem shows that the optimized minimum state LSA for a given one is uniquely determined.

Theorem 7 (**Minimization of Length of Threshold Sequences).** Let $M = (d, Q, q_0, F, w, h, s, \delta)$ and $M' = (d, Q', q'_0, F', w', h', s', \delta')$ be LSAs. If M is isomorphic to M' w.r.t. the isomorphism f, then $H(q) = H(f(q))$ holds for any $q \in Q$.

□

Now, we can say that the optimized minimum state LSA for a given one is uniquely determined because some optimized minimum state LSAs have the same weight function and the set of threshold values at every corresponding state.

In order to optimize an LSA $M = (d, Q, q_0, F, w, h, s, \delta)$, it is enough to remove all the threshold values $h(q)_i$ such that $s(q)_i = s(q)_{i+1}$ for any integer i and rewrite δ according to such changes. Let k be the maximum number of edges going out from $q \in Q$, and let $n = |Q|$. This procedure can be done in time $O(kn)$.

Example 3. The optimized version of the LSA in Figure 5 is illustrated in Figure 6.

□

6. Conclusions

In this paper, we presented an algorithm to minimize an LSA M. Its time complexity is $O((K + k)\, n^2)$, where K is the maximum number of threshold values assigned to each weight, k is the maximum number of edges going out from a state of M, and n is the number of states in M. We moreover discussed on the minimization of the length of a threshold sequence at each state.

Some algorithms to learn an original finite automaton uses a minimization algorithm as in 2), 9). Therefore the algorithm to minimize an LSA in this paper will play an important role in the theory of learning an LSA. The development of the theory of learning an LSA is one of the important future works.

References

1) Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T.A., Ho, P.-H., Nicollin, X., Olivero, A., Sifakis, J. and Yovine, S.: The Algorithmic Analysis of Hybrid Systems, *Theoretical Computer Science*, Vol.138, No.1, pp.3–34 (1995).
2) Dupont, P.: Incremental Regular Inference, *Proceedings of the Third International Colloquium on Grammatical Inference (ICGI-96)*, pp.222–237 (1996).
3) Lynch, N. and Vaandrager, F.: Forward and Backward Simulations for Timing-Based Systems, *Proceedings of the Real-Time: Theory in Practice*, pp. 397–446 (1992).
4) Matsunaga, T. and Oshita, M.: Recognition of Walking Motion Using Support Vector Machine, *ISICE2007*, pp.337–342 (2007).
5) Matsunaga, T. and Oshita, M.: Automatic estimation of motion state for motion recognition using SVM, *IPSJ SIG Technical Report*, Vol. 2008-CG-133, pp. 31–36 (2008).
6) Mohri, T. and Tanaka, H.: Weather Prediction by Memory-Based Reasoning, *Journal of Japanese Society for Artificial Intelligence*, Vol.10, No.5, pp.798–805 (1995).
7) N. Lynch, F. V.: Hybrid I/O automata, *Information and Computation*, Vol. 185, No.1, pp.103–157 (2003).
8) Numai, Y., Udagawa, Y. and Kobayashi, S.: Theory of Minimizing Linear Separation Automata, *IPSJ SIG Technical Report*, Vol.2009-MPS-76, pp.1–8 (2009).
9) Oncina, J. and García, P.: Inferring Regular Languages in Polynomial Updated Time, *Pattern Recognition and Image Analysis, Series in Machine Perception & Artificial Intelligence*, Vol.1, pp.49–61 (1992).
10) Yamato, J., Ohya, J. and Ishii, K.: Recognizing human action in time-sequential images using hidden Markov model, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp.379–385 (1992).