IPSJ Transactions on Mathematical Modeling and Its Applications Vol. 3 No. 3 67–79 (Oct. 2010)

Regular Paper

State and Threshold Sequence 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 the number of states 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 and a threshold sequence at every state, which determine the transition from the current state to the next 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 the minimization 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),2)] and a timed automaton [3)]. Many researchers utilize computational models that can deal with real values to solve various problems including weather forecasting [4)], motion recognition [5),6)], and time-sequential image analysis [7)]. Therefore, we believe that the establishment of the theory of automata that can deal with real values is very important.

The study of a state minimization algorithm is important at various research fields. For instance, in circuit design, the state minimization of finite sequential machine plays an essential role to design a minimum size circuit. In the theory of timed automata, a state minimization algorithm is utilized to improve the efficiency of the analysis of real-time systems [8)].

†1 Department of Computer Science, Graduate School of Electro-Communications, University of Electro-Communications
†2 NTT-IT Corporation

In our previous paper [9)], we theoretically analyzed a **linear separation automaton (LSA)**. An LSA accepts a sequence of real vectors, and has a weight function and a threshold sequence at every state, which determine the transition from the current state to the next at each step. Some algorithms to learn an original finite automaton are similar to state minimization algorithms as in Refs. 10), 11). Therefore the algorithm to minimize the number of states of an LSA in this paper will play an important role in the theory of learning an LSA.

We have established the theory of state minimization of a given LSA. The **state minimization** of an LSA M is to minimize the number of states of M. The state minimized LSA is termed the **minimum state** LSA. We proved Myhill-Nerode theorem for an LSA, established the uniqueness of the minimum state LSA for a given one, and characterized the minimum state LSA for a given one.

More precisely speaking, the characterization of the minimum state LSA is based on the concept of indistinguishability of states. We have proved that, if we merge all the indistinguishable pair of states of a given LSA, then we will obtain the minimum state version of a given one. However, we did not give any method to find a indistinguishable pair.

This paper gives a method to detect the indistinguishability of states, and 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 the minimization of a threshold sequence at each state.

In Section 2, we will give necessary definitions and notation needed in the sequel of this paper. Section 3 introduces a linear separation automaton (LSA) and some theoretical results. We present an algorithm to minimize the number of states of a given LSA in Section 4. In Section 5, we discuss the minimization of a threshold sequence at each state. Section 6 includes conclusions and future works.

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 $\mathbf{R}^d$. 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.

Let S be a set. A **partition** $\pi = \{S_1, \ldots, S_k\}$ of S is the set of mutually disjoint non-empty subsets S_i of S for $1 \le i \le k$ such that $\cup_{i=1,\ldots,k} S_i = S$. A partition $\pi = \{S_1, \ldots, S_k\}$ of $\mathbf{R}^d$ 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 $x \in \mathbf{R}^d$,

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

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

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

Proof. First, we will consider the case of $w \neq -w'$. Note that $(w \otimes w')^2 \neq 1$ holds. Let $c_1 = h$, $c_2 = h + \varepsilon$, and $h' = h$. Take

$$x_i = h\, w' + \alpha_i (w - (w \otimes w')\, w')$$

for $i = 1, 2$, where

$$\alpha_i = \frac{c_i - (w \otimes w')h}{1 - (w \otimes w')^2}.$$

Hence we have $w' \otimes x_1 = w' \otimes x_2 = h = h'$, $w \otimes x_1 = h$, and $w \otimes x_2 = h + \varepsilon$, which satisfies the claim.

Next, we will consider the case of $w = -w'$. Let $h' = -h$. Take

$$x_1 = hw \quad \text{and} \quad x_2 = (h + \varepsilon)w.$$

Hence we have $w' \otimes x_1 = -h = h'$, $w' \otimes x_2 = h' - \varepsilon$, $w \otimes x_1 = h$, and $w \otimes x_2 = h + \varepsilon$, which satisfies the claim. □

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 with respect to 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 [9].

In Section 3.1, we illustrate the state transition of an LSA. In Section 3.2, we formally define an LSA. We present some theoretical results for LSAs in Section 3.3.

3.1 Overview

An LSA is an extended model of a finite automaton. It accepts a sequence of real vectors, and has a weight and a threshold sequence at every state. The transition from the current state to the next 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.

Consider a state transition diagram as in Fig. 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 Fig. 1. The LSA M_1 has the weight function w, the threshold sequence function 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.

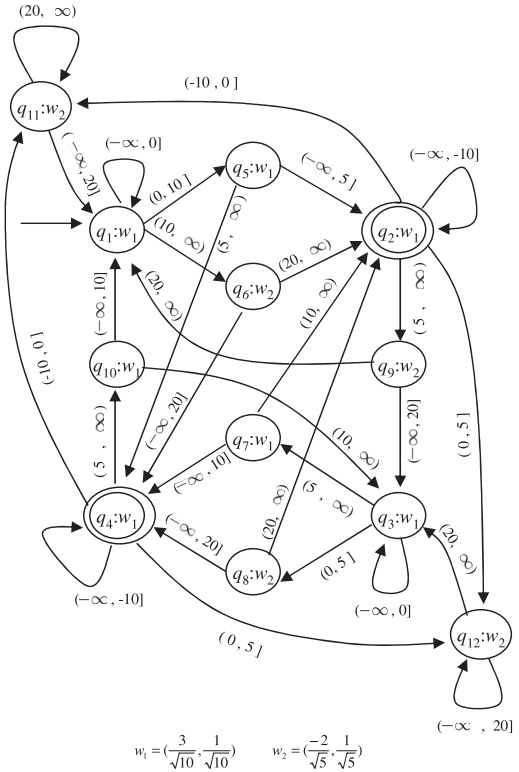


Fig. 1 LSA M_1.

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 sequence 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:

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

Consider a state transition diagram as in Fig. 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 $|Q|$ of 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 Ref. 9).

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:

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

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 $\equiv$.

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 LSAs. 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 $\mathbf{R}^d/\ R([\alpha]_{\approx_S})$. 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 $p \not\sim q$.

Lemma 2.

$$p \sim q \quad \Leftrightarrow \quad \forall \gamma \in (\mathbf{R}^d)^*,\ \delta(p, \gamma) \in F \textbf{ iff } \delta(q, \gamma) \in F.$$

Lemma 3.

$$p \sim q \quad \Leftrightarrow \quad \forall \alpha \in (\mathbf{R}^d)^*,\ \delta(p, \alpha) \sim \delta(q, \alpha).$$

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

$$Q' = Q/\sim, \quad q_0' = [q_0]_\sim, \quad F' = \{[q]_\sim \mid q \in F\},$$

$\delta'([q]_\sim, x) = [\delta(r(q), x)]_\sim, \quad w'([q]_\sim) = w(r(q)), \quad h'([q]_\sim) = h(r(q))$.

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. State Minimization Algorithm

In this section, we deal with an algorithm to minimize the number of states of a given LSA. This algorithm is similar to that to minimize the number of states of a given finite automaton [12),13)].

In Section 4.1, we illustrate the approach to minimize the number of states of an LSA, coarsest refinement approach. We next describe an algorithm to minimize the number of states of a given LSA in Section 4.2, present an example run of the algorithm in Section 4.3, and finally prove the correctness of the algorithm in Section 4.4.

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 4. 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 $\delta(q', x_1) = \delta(q', x_2)$.

Proof. Since $p_1 \prec_q p_2$, there exists an integer i such that $s(q)_{i-1} = p_1$ and $s(q)_i = p_2$. We deduce from Lemma 1 that there exists $h' \in \mathbf{R}$ such that, for any $\varepsilon > 0$, there exist $x_1, x_2 \in \mathbf{R}^d$ satisfying $h' - \varepsilon \le w(q') \otimes x_2 \le w(q') \otimes x_1 = h'$ and $h(q)_{i-1} = w(q) \otimes x_1 < w(q) \otimes x_2 \le h(q)_{i-1} + \varepsilon$. If we take such ε small enough, then we deduce from the definition of the state transition function δ that $\delta(q, x_i) = p_i$ $(i = 1, 2)$ and $\delta(q', x_1) = \delta(q', x_2)$, completing the proof. □

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:

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

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 Fig. 1. We have $W(Q) = \{w_1, w_2\}$, $H(w_1) = \{-10, 0, 5, 10, \infty\}$, and $H(w_2) = \{20, \infty\}$. Some functions in the set $\overline{\Delta}$ are represented in **Fig. 2** and **Fig. 3**.

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) $\forall B \in \pi\ \ (\ |W(B)| > 1\ \Rightarrow \exists B' \in \pi$ such that $\Delta(B) \subseteq B'\ )$.

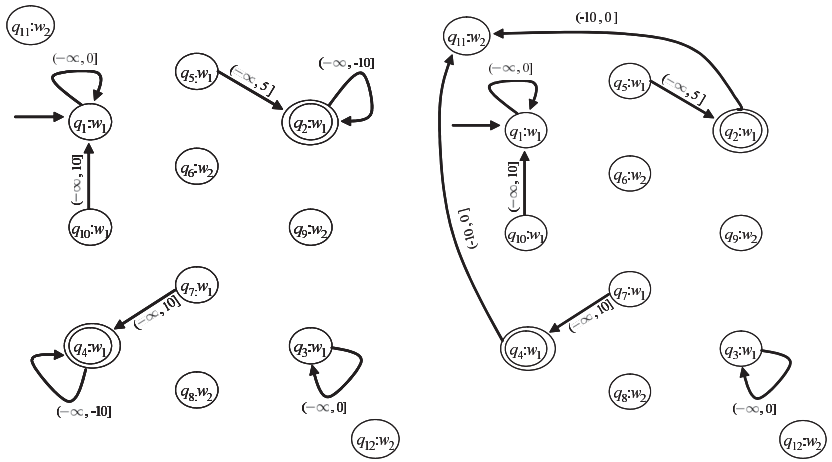
Proof. We first show that $Q/\sim$ is a refinement of π_0 satisfying **(C1)** and **(C2)**.

It is clear that $Q/\sim$ is a refinement of π_0, since we deduce from Lemma 2 that $q_1 \sim q_2$ implies $\delta(q_1, \lambda) \in F \Leftrightarrow \delta(q_2, \lambda) \in F$ implies $q_1 \in F \Leftrightarrow q_2 \in F$.

Assume that **(C1)** does not hold for $Q/\sim$, i.e., there exist $B \in Q/\sim$ and $f \in \overline{\Delta}$ such that $f(B) \not\subseteq B'$ for any $B' \in Q/\sim$. This implies that, from the definition of $\overline{\Delta}$, there exist $q_1, q_2 \in B$ and $x \in \mathbf{R}^d$ such that $q_1 \sim q_2$ and $\delta(q_1, x) \not\sim \delta(q_2, x)$. This contradicts Lemma 3. Therefore, the partition $Q/\sim$ satisfies **(C1)**.

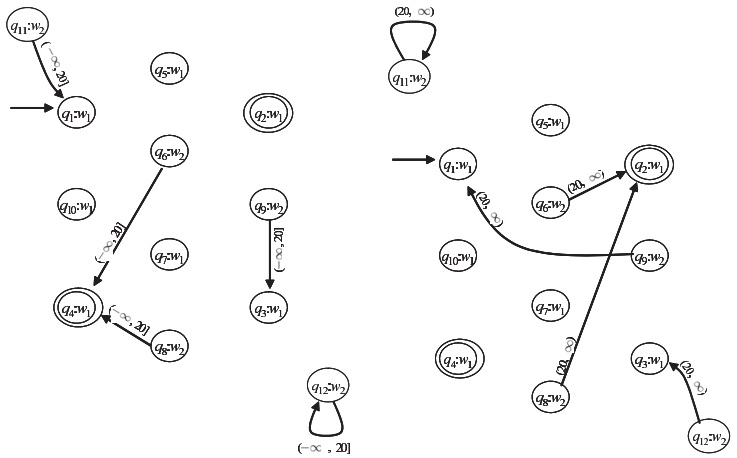
Assume that **(C2)** does not hold for $Q/\sim$, i.e., there exists $B \in Q/\sim$ such that $|W(B)| > 1$ and $\Delta(B) \not\subseteq B'$ for any $B' \in Q/\sim$. We consider the following two cases:

(Case 1) Consider the case that there exists $q \in B$ such that $\Delta(q) \not\subseteq B'$ for



$\delta_{w_1,-10}$ $\delta_{w_1,0}$

Fig. 2 Some graphs in $\overline{\Delta}$ for the weight w_1.



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

Fig. 3 Graphs in $\overline{\Delta}$ for the weight w_2.

any $B' \in Q/\sim$. There exist $p_1, p_2 \in Q$ such that $p_1 \prec_q p_2$ and $p_1 \not\sim p_2$. Let $q' \in B$ such that $q \not\sim_w q'$. Such q' should exist since $|W(B)| > 1$ holds. Lemma 4 implies that there exist $x_1, x_2 \in \mathbf{R}^d$ such that $\delta(q, x_i) = p_i$ $(i = 1, 2)$ and $\delta(q', x_1) = \delta(q', x_2)$. Since $p_1 \not\sim p_2$, we have $\delta(q, x_i) \not\sim \delta(q', x_i)$ for some $i = 1, 2$, which contradicts $q \sim q'$ from Lemma 3.

(Case 2) Consider the case that for any $q \in B$, there exists $B' \in Q/\sim$ such that $\Delta(q) \subseteq B'$. By the assumption that **(C2)** does not hold, there exist $q_1, q_2 \in B$ such that $\delta(q_1, x) \not\sim \delta(q_2, x)$ for any $x \in \mathbf{R}^d$, which contradicts $q_1 \sim q_2$ by Lemma 3.

In both cases, we have a contradiction. Thus **(C2)** holds for $Q/\sim$. In conclusion, $Q/\sim$ is a refinement of π_0 satisfying **(C1)** and **(C2)**.

Next we will show that $Q/\sim$ is the coarsest. Let π_* be any refinement of π_0 satisfying **(C1)** and **(C2)**. It suffices to show that for any $q_1, q_2 \in Q$, $q_1 \sim_{(\pi_*)} q_2$ implies $q_1 \sim q_2$.

Suppose that $q_1 \sim_{(\pi_*)} q_2$ and $q_1 \not\sim q_2$. Without loss of generality, we deduce from Lemma 2 that there exists $\alpha = \langle x_1, \ldots, x_n \rangle \in (\mathbf{R}^d)^*$ such that $\delta(q_1, \alpha) \in F$ and $\delta(q_2, \alpha) \notin F$. Let us define $q_i^{(j)} = \delta(q_i, \langle x_1, \ldots, x_j \rangle)$ for $i = 1, 2$ and $j = 0, \ldots, n$.

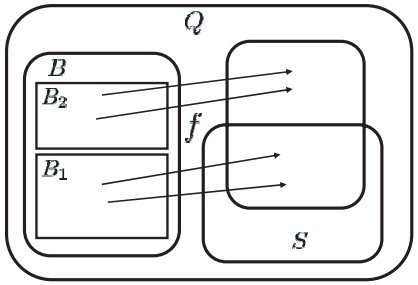
Note that $q_1^{(n)} \not\sim_{(\pi_*)} q_2^{(n)}$ holds since π_* is a refinement of π_0. We will show $q_1^{(n-1)} \not\sim_{(\pi_*)} q_2^{(n-1)}$. Assume $q_1^{(n-1)} \sim_{(\pi_*)} q_2^{(n-1)}$. In the case of $q_1^{(n-1)} \sim_w q_2^{(n-1)}$, take $\omega = w(q_1^{(n-1)})$ and $v = \min\{\, h \in H(\omega) \mid \omega \otimes x_n \leq h \,\}$. Let $f = \delta_{\omega,v} \in \overline{\Delta}$. We deduce from the condition **(C1)** with respect to f that $q_1^{(n)} \sim_{(\pi_*)} q_2^{(n)}$, which is a contradiction. In the case of $q_1^{(n-1)} \not\sim_w q_2^{(n-1)}$, we deduce from the condition **(C2)** that $q_1^{(n)} \sim_{(\pi_*)} q_2^{(n)}$, which is a contradiction. In both cases, we have a contradiction, and thus $q_1^{(n-1)} \not\sim_{(\pi_*)} q_2^{(n-1)}$ holds. By repeating the same discussion, we can prove that $q_1 = q_1^{(0)} \not\sim_{(\pi_*)} q_2^{(0)} = q_2$ holds, which is a contradiction.

Therefore, the assumption $q_1 \not\sim q_2$ is not correct. □

4.2 Minimization Algorithm

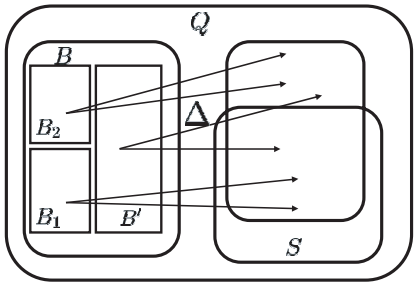
Our algorithm uses two primitive refinement operations $split_1$ and $split_2$; the former is for the condition **(C1)**, and the latter is for **(C2)**.

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



split B into B_1 and B_2
$(B \in \pi, S \subseteq Q, f \in \overline{\Delta})$

Fig. 4 $split_1(S, f, \pi)$.



split B into B_1, B_2, and B'_ω's
$(B \in \pi, S \subseteq Q, w \in W(B'))$

Fig. 5 $split_2(S, \pi)$.

defined as follows:

---

find all blocks $B \in \pi$ such that $f(B) \cap S \neq \emptyset$ and $f(B) \not\subseteq S$. Define $B_1 = B \cap f^{-1}(S)$ and $B_2 = B - B_1$. Then, 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 of Q, $split_2(S, \pi)$ is defined as follows:

---

find all blocks $B \in \pi$ such that $\Delta(B) \cap S \neq \emptyset$, $\Delta(B) \not\subseteq S$ and $|W(B)| > 1$, and split B into some smaller blocks defined in the following way; Define $B' = \{\, q \in B \mid \Delta(q) \cap S \neq \emptyset \text{ and } \Delta(q) \not\subseteq S \,\}$, $B_1 = \{q \in B - B' \mid \Delta(q) \subseteq S\}$, 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 **Fig. 4** and **Fig. 5**. A concrete example is shown in Section 4.3.

Now, we present an algorithm to minimize the number of states of 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

```
1: let π = {F, Q − F};
2: loop
3:    if ∃ B ∈ π such that split2(B, π) ≠ π then
4:       replace π with split2(B, π);
5:    else if ∃ B ∈ π, ∃ f ∈ Δ̄ such that split1(B, f, π) ≠ π then
6:       replace π with split1(B, f, π);
7:    else
8:       output π and halt;
9:    end if
10: end loop
```

How do we determine the weights and threshold sequences of the minimum state LSA? Recall that the definition of weights and threshold sequences in the minimum state LSA. Let $M = (d, Q, q_0, F, w, h, s, \delta)$ be an LSA. The LSA $M/\sim = (d, Q', q'_0, F', w', h', s', \delta')$ is the minimum state version of M. The weight w' and threshold sequence h' is defined as $w'([q]_\sim) = w(r(q))$ and $h'([q]_\sim) = h(r(q))$, respectively. The weight of the block $[q]_\sim$ is equal to $w(q')$ such that

$q' \sim q$. Similarly, The threshold sequence of the block $[q]_\sim$ is equal to $h(q')$ such that $q' \sim q$.

4.3 Example Run

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

Let $\pi = \{B_a, B_b\}$, where

$B_a = \{q_2, q_4\}$,

$B_b = \{q_1, q_3, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}\}$.

First, consider $split_2(B_a, \pi)$. We find that $\Delta(B_b) \cap B_a \neq \emptyset$, $\Delta(B_b) \not\subseteq B_a$, and $|W(B_b)| = 2 > 1$. Moreover, $B' = \emptyset$. Thus, we split B_b into $B_1 = \{\, q \in B_b - B' \mid \Delta(q) \subseteq B_a \,\} = \{q_5, q_6, q_7, q_8\}$ and $B_2 = (B_2 - B') - B_1 = \{q_1, q_3, q_9, q_{10}, q_{11}, q_{12}\}$. We rename B_1 and B_2 as B_c and B_d, respectively. Hence, the operation $split_2(B_a, \pi)$ constructs the new partition $\pi_1 = \{B_a, B_c, B_d\}$, where

$B_a = \{q_2, q_4\}$,

$B_c = \{q_5, q_6, q_7, q_8\}$,

$B_d = \{q_1, q_3, q_9, q_{10}, q_{11}, q_{12}\}$.

Next, we have $split_2(B_d, \pi_1) = \pi_2 = \{B_a, B_c, B_e, B_f\}$, where

$B_a = \{q_2, q_4\}$,

$B_c = \{q_5, q_6, q_7, q_8\}$,

$B_e = \{q_1, q_3\}$,

$B_f = \{q_9, q_{10}, q_{11}, q_{12}\}$.

Next, we have $split_2(B_e, \pi_2) = \pi_3 = \{B_a, B_c, B_e, B_g, B_h\}$, where

$B_a = \{q_2, q_4\}$,

$B_c = \{q_5, q_6, q_7, q_8\}$,

$B_e = \{q_1, q_3\}$,

$B_g = \{q_9, q_{10}\}$,

$B_h = \{q_{11}, q_{12}\}$.

Finally, we have $split_1(B_h, \delta_{w_2,20}, \pi_3) = \pi_4 = \{B_a, B_c, B_e, B_g, B_i, B_j\}$, where

$B_a = \{q_2, q_4\}$,

$B_c = \{q_5, q_6, q_7, q_8\}$,

$B_e = \{q_1, q_3\}$,

$B_g = \{q_9, q_{10}\}$,

$B_i = \{q_{11}\}$,

$B_j = \{q_{12}\}$.

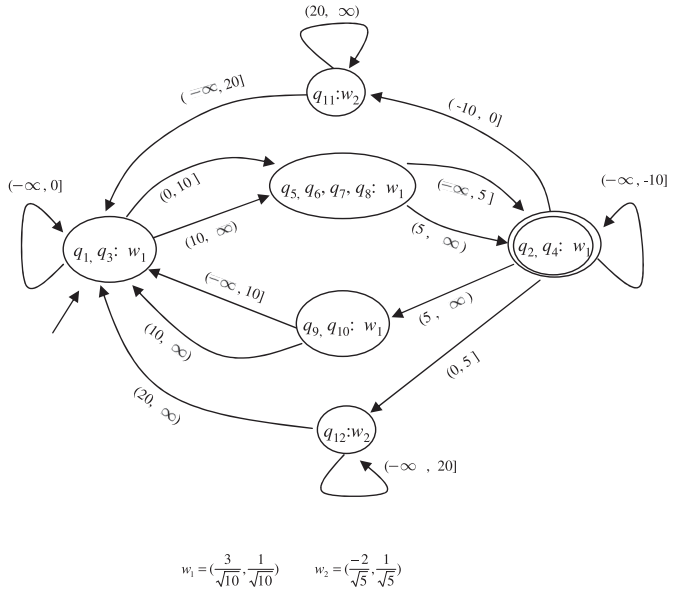


Fig. 6 Minimum state LSA of M_1.

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

Next, we determine the weight function and the threshold sequence function. The weight of B_a, $w(B_a)$, is $w(q_2) = w(q_4) = w_1$ because q_2 and q_4 are in B_a. The weight $w(B_c)$ is either $w(q_5) = w(q_7) = w_1$ or $w(q_6) = w(q_8) = w_2$. Which weight is correct? Of course, any weight is correct. The rest of weights are also determined in a similar way. The threshold sequence $h(B_a)$ is $h(q_2) = h(q_4) = \langle -10, 0, 5 \rangle$. The rest of threshold sequences are also determined in a similar way.

Finally, the minimum state LSA for M_1 with the set π_4 of states is obtained in **Fig. 6**.

4.4 Correctness of Algorithm

We give some basic properties of these operations:

Lemma 5. A partition π satisfies **(C1)** if and only if $split_1(B, f, \pi) = \pi$ for every block $B \in \pi$ and $f \in \overline{\Delta}$. A partition π satisfies **(C2)** if and only if $split_2(B, \pi) = \pi$ for every block $B \in \pi$.

Proof. It is clear from the definition of $split_1$ that the first statement of this

lemma holds.

First, we will prove that π satisfies **(C2)** only if $split_2(B,\pi)=\pi$ for every block $B\in\pi$. Suppose that π satisfies **(C2)** and $split_2(B,\pi)\neq\pi$ for some $B\in\pi$. There exists $B'\in\pi$ such that $\Delta(B')\cap B\neq\emptyset$, $\Delta(B')\not\subseteq B$ and $|w(B')|>1$, which is a contradiction.

Next, we will prove that π satisfies **(C2)** if $split_2(B,\pi)=\pi$ for every block $B\in\pi$. Suppose that $split_2(B,\pi)=\pi$ holds for every block $B\in\pi$ and π does not satisfy **(C2)**. There exists $B\in\pi$ such that $|W(B)|>1$ and for any $B'\in\pi$, $\Delta(B)\not\subseteq B'$ holds. Take a block $B_*\in\pi$ such that B_* has an element of $\Delta(B)$. We have $\Delta(B)\cap B_*\neq\emptyset$ and $\Delta(B)\not\subseteq B_*$, which implies that $split_2(B_*,\pi)\neq\pi$, which is a contradiction. □

Lemma 6. 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.

Proof. We will prove the first statement. Let π_1 be a partition satisfying $split_1(S,f,\pi_1)=\pi_1$ and π_2 be a refinement of π_1. Assume that $split_1(S,f,\pi_2)\neq\pi_2$ holds. Then, there exists a block $B\in\pi_2$ such that $f(B)\cap S\neq\emptyset$ and $f(B)\not\subseteq S$. Since π_2 is a refinement of π_1, there exists a block $B'\in\pi_1$ such that $B\subseteq B'$. Then, $f(B')\cap S\neq\emptyset$ and $f(B')-S\supseteq f(B)-S\neq\emptyset$ hold. Therefore, $split_1(S,f,\pi_1)\neq\pi_1$ holds, which is a contradiction.

The second statement of this lemma can be proved in a similar manner. □

Lemma 7. 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 $split_2(S_1\cup S_2,\pi)=\pi$.

Proof. We will prove the first statement. Consider a partition π such that $split_1(S_1,\ f,\pi)=\pi$ and $split_1(S_2,f,\pi)=\pi$ hold. Assume that $split_1(S_1\cup S_2,f,\pi)\neq\pi$ holds. Then, there exists a block $B\in\pi$ such that $f(B)\cap(S_1\cup S_2)\neq\emptyset$ and $f(B)\not\subseteq(S_1\cup S_2)$ hold. We deduce from $f(B)\cap(S_1\cup S_2)\neq\emptyset$ that either $f(B)\cap S_1\neq\emptyset$ or $f(B)\cap S_2\neq\emptyset$ holds. We may assume $f(B)\cap S_1\neq\emptyset$ without loss of generality. The equation $f(B)\not\subseteq(S_1\cup S_2)$ implies $f(B)\not\subseteq S_1$. Therefore, $split_1(S_1,f,\pi)\neq\pi$ holds, which is a contradiction.

The second statement can be proved in a similar way. □

Lemma 8. 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 $split_1(S,f,\pi_2)$.

Proof. Let B be any block in π_1. There exists $B'\in\pi_2$ such that $B\subseteq B'$ since π_1 is a refinement of π_2. The operation $split_1(S,f,\pi_1)$ might split $B\in\pi_1$ into $B_1=f^{-1}(S)\cap B$ and $B_2=B-B_1$. The operation $split_1(S,f,\pi_2)$ might split $B'\in\pi_2$ into $B'_1=f^{-1}(S)\cap B'$ and $B'_2=B'-B'_1$. Then, we have $B_1\subseteq B'_1$ and $B_2=B-B_1=B-f^{-1}(S)\subseteq B'-f^{-1}(S)=B'-B'_1=B'_2$. Therefore, in the case that B is split into B_1 and B_2 by $split_1(S,f,\pi_1)$, whether B' is split or not, for each $i=1,2$, there exists a block $D_i\in split_1(S,f,\pi_2)$ with $B_i\subseteq D_i$. Thus, it is left to consider the case that B is not split, but B' is split into B'_1 and B'_2.

Recall that $split_2(S,\pi_2)=\pi_2$ holds. Therefore, for any $D\in\pi_2$ such that $\Delta(D)\cap S\neq\emptyset$ and $\Delta(D)\not\subseteq S$, we have $|W(D)|=1$. From the assumption that B' is split, $|W(B')|=1$ should hold. Thus, $W(B)=W(B')=\{\omega\}$ for some $\omega\in W(Q)$. This implies that f is defined either on every $q\in B'$, or on no $q\in B'$.

Since B is not split by $split(S,f,\pi_1)$, we have either $f(B)\subseteq S$ or $f(B)\cap S=\emptyset$.

In the case of $f(B)\cap S=\emptyset$, we have $f^{-1}(S)\cap B=\emptyset$, which implies $B'_1\cap B=f^{-1}(S)\cap B'\cap B=\emptyset$. Therefore, we deduce from $B\subseteq B'$ that $B\subseteq B'-B'_1=B'_2$ holds.

Let us assume $f(B)\subseteq S$. In the case that f is defined on no $q\in B'$, $f(B)\cap S=\emptyset$ holds. Thus, by the discussion of the previous paragraph, we have $B\subseteq B'_2$. Therefore, we should consider the case that f is defined on every $q\in B'$. In this case, we deduce from the assumption $f(B)\subseteq S$ that $B\subseteq f^{-1}(S)$, which implies $B\subseteq f^{-1}(S)\cap B\subseteq f^{-1}(S)\cap B'=B'_1$.

In conclusion, for every block in $split_1(S,f,\pi_1)$, there exists a block in $split_1(S,f,\pi_2)$ containing it. Thus, $split_1(S,f,\pi_1)$ is a refinement of $split_1(S,f,\pi_2)$. □

Lemma 9. 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 $split_2(S,\pi_2)$.

Proof. Let B be any block in π_1. There exists $D\in\pi_2$ such that $B\subseteq D$ since π_1 is a refinement of π_2. Let $B'=\{q\in B\mid\Delta(q)\cap S\neq\emptyset,\Delta(q)\not\subseteq S\}$, $B_1=\{q\in B-B'\mid\Delta(q)\subseteq S\}$ and $B_2=(B-B')-B_1$. The operation $split_2(S,\pi_1)$ might split B into B_1, B_2, and B'_ω's ($\omega\in W(B')$). On the other hand, let

$D' = \{q \in D \mid \Delta(q) \cap S \neq \emptyset, \Delta(q) \not\subseteq S\}$, $D_1 = \{q \in D - D' \mid \Delta(q) \subseteq S\}$ and $D_2 = (D - D') - D_1$. The operation $split_2(S, \pi_2)$ might split D into D_1, D_2, and D'_ω's ($\omega \in W(D')$).

Now, we will prove that $B' \subseteq D'$, $B_1 \subseteq D_1$, and $B_2 \subseteq D_2$.

It is clear that $B' \subseteq D'$, because $B \cap D' = B'$ holds.

We have

$$\begin{aligned} D_1 \cap B &= \{q \in D - D' \mid \Delta(q) \subseteq S\} \cap B \\ &= \{q \in D \mid \Delta(q) \subseteq S\} \cap B \\ &= \{q \in B \mid \Delta(q) \subseteq S\} \\ &= \{q \in B - B' \mid \Delta(q) \subseteq S\} \\ &= B_1. \end{aligned}$$

Hence $B_1 \subseteq D_1$ holds.

Furthermore, we have

$$\begin{aligned} B_2 &= (B - B') - B_1 \\ &= (B - B \cap D') - B \cap D_1 \\ &= (B - D') - D_1 \\ &\subseteq (D - D') - D_1 \\ &= D_2. \end{aligned}$$

Therefore, in the case that B is split by $split_2(S, \pi_1)$, whether D is split or not, for any of B_1, B_2 and B'_ω's, there exists a block in $split_2(S, \pi_2)$ containing it. Thus, it is left to consider the case that B is not split, but D is split.

Since B is not split, we have either $\Delta(B) \cap S = \emptyset$, $\Delta(B) \subseteq S$, or $|W(B)| = 1$.

In the case of $\Delta(B) \cap S = \emptyset$, we deduce from the definition of D' and D_1 that $D' \cap B = \emptyset$ and $B \cap D_1 = \emptyset$. Therefore, $B \subseteq (D - D') - D_1 = D_2$ holds.

In the case of $\Delta(B) \subseteq S$, we deduce from the definition of D' that $D' \cap B = \emptyset$, which implies $B \subseteq D - D'$. Therefore, from the definition of D_1, we have $B \subseteq D_1$.

Finally, let us consider the case of $|W(B)| = 1$. Recall that π_1 satisfies **(C1)**. Let $W(B) = \{\omega\}$. We consider the following two subcases: (a) $D' \cap B \neq \emptyset$ and (b) $D' \cap B = \emptyset$.

Subcase (a): There exists $q \in B$ such that $\Delta(q) \cap S \neq \emptyset$ and $\Delta(q) \not\subseteq S$. Then, there should exist $v_1, v_2 \in H(\omega)$ such that $\delta_{\omega,v_1}(q) \in S$ and $\delta_{\omega,v_2}(q) \notin S$. Since π_1 satisfies **(C1)** and S is a union of blocks in π_1, we have $\delta_{\omega,v_1}(q') \in S$ and $\delta_{\omega,v_2}(q') \notin S$ for any $q' \in B$. Therefore, $B \subseteq D'$ holds, which implies that $B \subseteq D'_\omega$.

Subcase (b): For any $q \in B$, either $\Delta(q) \cap S = \emptyset$ or $\Delta(q) \subseteq S$ holds. Let q be any element of B. In the case of $\Delta(q) \cap S = \emptyset$, since π_1 satisfies **(C1)** and S is a union of blocks in π_1, we have $\Delta(q') \cap S = \emptyset$ for any $q' \in B$, which implies $B \subseteq D_2$. In the case of $\Delta(q) \subseteq S$, since π_1 satisfies **(C1)** and S is a union of blocks in π_1, we have $\Delta(q') \subseteq S$ for any $q' \in B$, which implies that $B \subseteq D_1$.

In conclusion, for every block in $split_2(S, \pi_1)$, there exists a block in $split_2(S, \pi_2)$ containing it. Thus, $split_2(S, \pi_1)$ is a refinement of $split_2(S, \pi_2)$. □

Lemma 10. 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.

Proof. By induction on the number of refinement steps. The claim is true initially from the definition. Let π_* be the coarsest refinement of $\{F, Q - F\}$ satisfying **(C1)** and **(C2)**. We deduce from Lemma 5 that $split_1(B, f, \pi_*) = \pi_*$ and $split_2(B, \pi_*) = \pi_*$ for any $B \in \pi_*$ and $f \in \overline{\Delta}$.

Suppose that the claim holds before a refinement step by $split_1(B', f, \pi)$ or $split_2(B', \pi)$ for some $B' \in \pi$. Since π_* is a refinement of π by the induction hypothesis, B' is a union of some blocks in π_*. Consider the case that the refinement is done by $split_1(B', f, \pi)$. Note that in this case $split_2(B', \pi) = \pi$ holds by the structure of if-conditions of Algorithm 1. Lemma 7 implies that $split_1(B', f, \pi_*) = \pi_*$. Therefore Lemma 8 implies that $\pi_* = split_1(B', f, \pi_*)$ is a refinement of $split_1(B', f, \pi)$.

In the case that the refinement is done by $split_2(B, \pi)$, we can prove the induction step in a similar way by using Lemma 9. □

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 5 implies that the final partition π_f satisfies **(C1)** and **(C2)**. Moreover, Lemma 10 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 Threshold Sequences

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

In this section, we will elucidate some important properties of LSAs related to the threshold sequence, and minimize 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.

The **threshold sequence minimization** of an LSA M is to remove all the unnecessary threshold values at all the states of M.

Lemma 11. 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' with respect to the isomorphism f, then $w(q) = w'(f(q))$ holds for any $q \in Q$ such that $|\Delta(q)| > 1$.

Proof. For any $q \in Q$ such that $|\Delta(q)| > 1$, there exists an integer i such that $s(q)_{i-1} \neq s(q)_i$.

Now, assume that $w(q) \neq w'(f(q))$. We deduce from Lemma 1 that there exists $h' \in \mathbf{R}$ such that, for any $\varepsilon > 0$, there exist $x_1, x_2 \in \mathbf{R}^d$ satisfying $h' - \varepsilon \leq w'(f(q)) \otimes x_2 \leq w'(f(q)) \otimes x_1 = h'$ and $h(q)_{i-1} = w(q) \otimes x_1 < w(q) \otimes x_2 \leq h(q)_{i-1} + \varepsilon$. If we take such ε small enough, then we deduce from the definition of the state transition function δ that $\delta(q, x_1) = s(q)_{i-1}$ and $\delta(q, x_2) = s(q)_i$, and $\delta'(f(q), x_1) = \delta'(f(q), x_2)$.

The equation $\delta(q, x_1) \neq \delta(q, x_2)$ holds since $s(q)_{i-1} \neq s(q)_i$. Hence $f(\delta(q, x_1)) \neq f(\delta(q, x_2))$ holds since f is injective. We deduce $\delta'(f(q), x_1) \neq \delta'(f(q), x_2)$ from the definition of f, which is a contradiction. □

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 state and threshold sequence minimized LSA for a given one is uniquely determined.

Theorem 7 (**Minimization 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' with respect to the isomorphism f, then $H(q) = H(f(q))$ holds for any $q \in Q$.

Proof. Consider any $q \in Q$.

If $|\Delta(q)| = 1$, then it is clear that $H(q) = H(f(q)) = \emptyset$.

Suppose that $|\Delta(q)| > 1$ and consider any $h \in H(q)$. We will prove the following claim. The claim below immediately implies $h \in H(f(q))$, i.e., $H(q) \subseteq H(f(q))$.

Claim (A): For any $\varepsilon > 0$, there exists $h' \in H(f(q))$ such that $|h - h'| < \varepsilon$.

Let ε be any positive real. There exists an integer i such that $h(q)_i = h$ and $s(q)_{i-1} \neq s(q)_i$. Let $\varepsilon' = \min\{\varepsilon/2, h(q)_{i+1} - h(q)_i, h(q)_i - h(q)_{i-1}\}$ and take any $x, y \in \mathbf{R}^d$ such that $h - \varepsilon' < w(q) \otimes x < h < w(q) \otimes y < h + \varepsilon'$. We deduce from the definition of ε' that $h(q)_{i-1} < w(q) \otimes x < h(q)_i < w(q) \otimes y < h(q)_{i+1}$. We have

$$\begin{aligned} \delta(q, x) \neq \delta(q, y) &\Rightarrow f(\delta(q, x)) \neq f(\delta(q, y)) \quad \text{(since } f \text{ is injective)} \\ &\Rightarrow \delta'(f(q), x) \neq \delta'(f(q), y). \quad \text{(from the definition of } f\text{)} \end{aligned}$$

Therefore, there exists $h' \in H(f(q))$ satisfying either (a) $w'(f(q)) \otimes x \leq h' < w'(f(q)) \otimes y$ or (b) $w'(f(q)) \otimes y \leq h' < w'(f(q)) \otimes x$. Note that $w'(q) = w(f(q))$

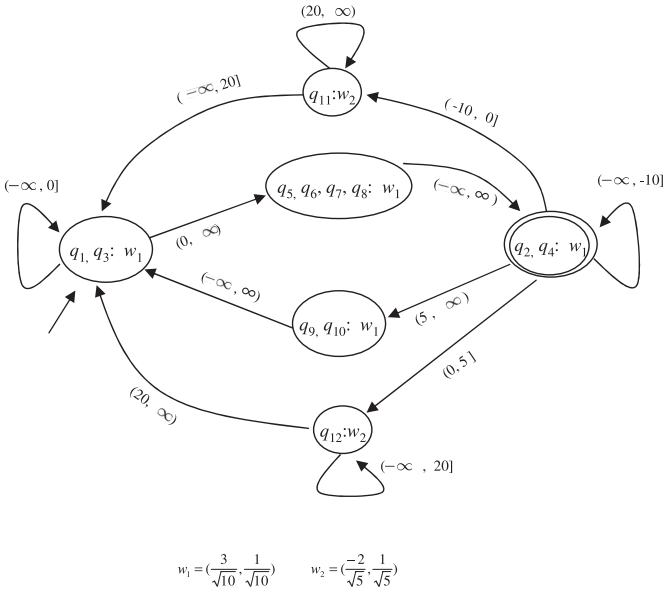


Fig. 7 Threshold sequence minimized version of LSA in Fig. 6.

holds from Lemma 11. This means that $w'(f(q)) \otimes x < w'(f(q)) \otimes y$, and thus the inequality (a) only holds. Therefore it holds that $w(q) \otimes x \leq h' < w(q) \otimes y$. Finally, we have

$$\begin{aligned} |h - h'| &\leq w(q) \otimes (y - x) \quad (\text{since } h < w(q) \otimes y \text{ and } w(q) \otimes x \leq h') \\ &< 2\varepsilon' \quad (\text{from the conditions of } x \text{ and } y) \\ &\leq \varepsilon, \quad (\text{since } \varepsilon' \leq \varepsilon/2) \end{aligned}$$

completing the proof of **Claim (A)**.

In a similar manner, we can prove $H(f(q)) \subseteq H(q)$. □

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

In order to minimize the threshold sequences of a given 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 threshold sequence minimized version of the LSA in Fig. 6 is illustrated in **Fig. 7**.

6. Conclusions

In this paper, we presented 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 discussed the minimization of a threshold sequence at each state.

We minimized the number of states of a given LSA with coarsest refinement approach. There is another approach for minimization, state merging approach. This approach merges the indistinguishable pair of states of a given LSA one by one to obtain the minimum state LSA. To theorize this approach and compare two approaches is one of the open problems.

There exists a faster state minimization algorithm than that for the original finite automata [14),15)]. Therefore, we might theorize a faster state minimization algorithm than that in this paper. To theorize that algorithm is another open problem.

Some algorithms to learn an original finite automaton is similar to state minimization algorithms as in Refs. 10), 11). Therefore the algorithm to minimize the number of states of 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, *Theor. Comput. Sci.*, Vol.138, No.1, pp.3–34 (1995).
2) Lynch, N. and Vaandrager, F.: Hybrid I/O automata, *Information and Computation*, Vol.185, No.1, pp.103–157 (2003).
3) Lynch, N. and Vaandrager, F.: Forward and Backward Simulations for Timing-Based Systems, *Proc. Real-Time: Theory in Practice*, pp.397–446 (1992).
4) Mohri, T. and Tanaka, H.: Weather Prediction by Memory-Based Reasoning, *Jour-*

nal of Japanese Society for Artificial Intelligence, Vol.10, No.5, pp.798–805 (1995).
5) Matsunaga, T. and Oshita, M.: Recognition of Walking Motion Using Support Vector Machine, *ISICE2007*, pp.337–342 (2007).
6) 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).
7) Yamato, J., Ohya, J. and Ishii, K.: Recognizing human action in time-sequential images using hidden Markov model, *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp.379–385 (1992).
8) Alur, R., Courcoubetis, C., Halbwachs, N., Dill, D.L. and Wong-Toi, H.: Minimization of Timed Transition Systems, *CONCUR '92: Proc. Third International Conference on Concurrency Theory*, pp.340–354 (1992).
9) Numai, Y., Udagawa, Y. and Kobayashi, S.: Theory of Minimizing Linear Separation Automata, *IPSJ Transactions on Mathematical Modeling and Its Applications*, Vol.3, No.2, pp.83–91 (2010).
10) Dupont, P.: Incremental Regular Inference, *Proc. Third International Colloquium on Grammatical Inference* (*ICGI-96*), pp.222–237 (1996).
11) 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).
12) Harrison, M.A.: *Introduction to Switching and Automata Theory*, McGraw-Hill (1965).
13) McCluskey, E.J.: *Introduction to the Theory of Switching Circuits*, McGraw-Hill (1965).
14) Hopcroft, J.E.: An $n \log n$ algorithm for minimizing states in a finite automaton, *Theory of Machines and Computations*, pp.189–196 (1971).
15) Paige, R. and Tarjan, R.E.: Three Partition Refinement Algorithms, *SIAM Journal on Computing*, Vol.16, No.6, pp.973–989 (1987).

(Received February 4, 2010)
(Revised March 24, 2010)
(Accepted May 12, 2010)

Yuji Numai was born in 1984. He received his B.E. and M.E. degrees from University of Electro-Communications (UEC) in 2006 and 2008, respectively. He has been a doctoral student at UEC since 2008. His current research interests are the theory of formal languages and its learning.

Yoshiaki Udagawa was born in 1984. He graduated from University of Electro-Communications in 2007. He has been working in NTT-IT Corporation since 2007, and is now a developer of the Web Conference System Division of NTT-IT.

Satoshi Kobayashi was born in 1965. He received his B.E., M.E., and D.E. degrees from the University of Tokyo in 1988, 1990, and 1993, respectively. Since 2007 he has been a professor of Department of Computer Science, University of Electro-Communications. His research interests include computational learning theory, theory of molecular computing, bioinformatics. He is a member of IPSJ, IEICE, and JSAI.