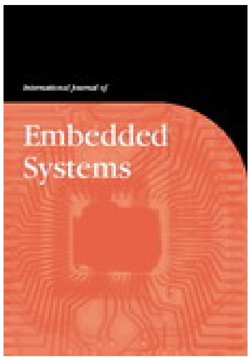


International Journal of Embedded Systems

ISSN online: 1741-1076 - ISSN print: 1741-1068
https://www.inderscience.com/ijes

Pin conjecture for matching automata

Zihao Wang, Qingxia Long, Yong He

DOI: 10.1504/IJES.2024.10068672

Article History:

Received:	27 October 2023
Last revised:	23 February 2024
Accepted:	18 March 2024
Published online:	10 February 2025

Pin conjecture for matching automata

Zihao Wang, Qingxia Long and Yong He*

School of Computer Science and Engineering,
Hunan University of Science and Technology,
Xiangtan, Hunan, China
Email: zhwang0112@outlook.com
Email: qingxialong77@gmail.com
Email: yonghe@hnust.edu.cn
*Corresponding author

Abstract: Let $\mathscr{A} = (S, A, \delta)$ be an n-state automaton over m input letters. The merging graph of $\mathscr{A}$ is the simple undirected graph $\mathbf{G} = (S, E)$ of which the edge set E consists of all binary mergable sets of $\mathscr{A}$. The automaton $\mathscr{A}$ is said to be matching if E is a matching of the state set S. The matchingness of $\mathscr{A}$ can be decided in $O(nm)$ times. If $\mathscr{A}$ is a matching automaton, then the rank of $\mathscr{A}$ is $n/2$ and, for any integer r with $n/2 \leq r \leq n$, an r-rank input word of $\mathscr{A}$ having length at most $\sum_{i=0}^{n-r} i$ can be computed in $O(n^2 + nm)$ times. This illustrates that Pin conjecture (and hence rank conjecture) is true for matching automata. Furthermore, it is shown that the tight bound for the lengths of the shortest r-rank input words of n-state matching automata is $\sum_{i=0}^{n-r} i$.

Keywords: r-rank input word; Pin conjecture; merging graph; matching automaton; extreme matching automaton.

Reference to this paper should be made as follows: Wang, Z., Long, Q. and He, Y. (2024) 'Pin conjecture for matching automata', *Int. J. Embedded Systems*, Vol. 17, Nos. 3/4, pp.224–236.

Biographical notes: Zihao Wang received his Bachelor of Engineering in Intelligent Equipment Academy from Shandong University of Science and Technology in 2020. He is currently working toward his Master of Engineering in Department of Computer Science and Technology at Hunan University of Science and Technology. His research interests include formal language and automata theory, combination theory of words and de Bruijn words.

Qingxia Long received her Bachelor of Engineering in Xiaoxiang Academy from Hunan University of Science and Technology in 2023. She is currently working toward her Master of Engineering in Department of Network Security at Hunan University of Science and Technology. Her research interests include formal language and automata theory and the theory of algorithm.

Yong He is a Professor of School of Computer Science and Engineering at Hunan University of Science and Technology. He received his Master of Science from South China Normal University in 1994 and Doctor of Science from Sun Yat-sen University in 2002. His research interests include formal language and automata theory, code theory and formal linguistics.

1 Introduction

Automaton is the simplest computational model served as the mathematical model of many discrete control systems such as computers and relay control systems. An automaton consists of a state set, an input alphabet and a state transition function that corresponds to the set of states, the set of input orders and the state transition rule of a discrete control system, respectively, The input words of an automaton are the finite sequences of input letters corresponding to the input order sequences of a discrete control system.

Synchronising automaton is a class of special automata raised from the consideration on the following typical problem in control theory (Ashby, 1956): how can we restore control over such a device if we do not know its current state but can observe outputs produced by the device under various actions? An automaton is synchronising if it has a synchronising word (i.e., an input word that can transition all states to the same state). Clearly, the synchronising words are helpful with restoreing control over the devices that can be described as a synchronising automaton. Synchronising automata were invented by Černý (Černý, 1964) in 1964, and re-invented several times by the researchers from different categories such as electronics, industrial automation and robotics. Since 1990s, due to the significant applications of synchronising automata in robotics (Eppstein, 1990), part handling problem in industrial automation (Ashby, 1956), communication (Burgin et al., 2001; Jürgensen, 2008), biocomputing (Benenson et al., 2001; Benenson, 2003), multiple valued logic and symbolic dynamics (Hrishnaswamy et al., 2008; Salomaa, 2012), many researches had been involved in the study on synchronising automata. The main works are revolved around the following *Černý conjecture* raised by Černý (Černý, 1964) in 1964: every n-state synchronising automaton has a synchronising word of length not exceeding $(n-1)^2$. Till now, this conjecture is still open except for some special cases (Almeida et al., 2009; Cui et al., 2019, 2023; de Bondt et al., 2020; Dubuc, 1998; Eppstein, 1990; He et al., 2021; Kari, 2003; Pin, 1983; Steinberg, 2011; Trahtman, 2007; Volkov, 2009, 2008), and it has become the longest-standing open problem in combinatorial theory on automata.

To consider the problem of restoring control over the devices that can not be described as a synchronising automaton, the notion *rank* for input words as well as automata were advised and investigated. An input word w of an automaton is r-rank if the states can be partitioned into r classes such that w is capable to transition the states in the same class to the same states. An automaton is r-rank if r is the minimum rank of its input words. Then the synchronising words are exactly the 1-rank input words, while synchronising automata are exactly the 1-rank automata. As a generalisation of Černý conjecture, Pin raised the following *Pin conjecture* in 1978: if an n-state automaton has an r-rank input word, then it has an r-rank input word of length not exceeding $(n-r)^2$. Pin conjecture was confirmed for many cases (Ananichev and Volkov, 2004, 2005; Imreh and Steinby, 1964; Pin, 1978a,b), but negated by Kari (2001) for the case that $n-r=4$. The restriction of Pin conjecture on the input words having the minimum rank (which are called the terminal input words) is named by *rank conjecture* (Almeida and Steinberg, 2009; Rystsov, 1992). This restricted conjecture has neither been proved nor disproved.

The aim of this paper is to introduce matching automata and confirm Pin conjecture (and hence rank conjecture) for matching automata. The needed terminologies and notations are explained in Section 2. The definition and the decision algorithm of matching automata are given in Section 3. In Section 4, Pin conjecture is confirmed for matching automata. The tight bound for the lengths of the shortest r-rank input words of matching automata are determined in Section 5. The last section consists of a brief summary and some discussions.

2 Preliminaries

2.1 The matchings of sets

The cardinality of a set X is denoted by $|X|$. The Cartesian product of two sets X and Y is denoted by $X \times Y$, i.e.,

$$X \times Y = \{(x,y) : x \in X,\, y \in Y\}.$$

A *partition* of a set X is a family $\{X_i : i \in I\}$ of pairwise disjoint non-empty subsets of X satisfying the condition that $X = \bigcup_{i\in I} X_i$. Every subset in a partition of X is called a *class*. A *matching* of a set X is a partition of X in which every class is a binary subset of X.

A *simple undirected graph* is an ordered pair of the form $\mathbf{G} = (V, E)$ in which V is a finite set called the *vertex set*, and E is a family of binary subsets of V called the *edge set*. The members of V and E are called the *vertices* and the *edges* of $\mathbf{G}$, respectively. The simple undirected graph $\mathbf{G} = (V, E)$ is called a *complete graph* if E consists of all binary subsets of V, and it is called a 1*-regular graph* if E is a matching of V.

2.2 Words and free monoids

Given an alphabet A. The finite sequences over A are called *words*. The empty sequence is called the *empty word* and denoted by ϵ. A non-empty word $a_1, a_2, \ldots, a_n$ is usually written as $a_1a_2\cdots a_n$. The length of a word w is denoted by $\ell(w)$. The *concatenation* wu of a word $w = a_1a_2\cdots a_n$ by a word $u = b_1b_2\cdots b_m$ is the word $a_1a_2\cdots a_nb_1b_2\cdots b_m$. Then

$$\ell(wu) = \ell(w) + \ell(u).$$

The set of all words over the alphabet A is denoted by A^*. The algebraic structure of the set A^* with respected to the concatenation operation is called the *free monoid* over A. A word u is a *factor* of a word w if there exist two words v and v' such that $w = vuv'$. Specially, a word u is a *left factor* of a word w if there exists some word v' such that $w = uv'$.

2.3 Automata

An *automaton* is a triad of the form $\mathscr{A} = (S, A, \delta)$ in which S is a finite set called the *state set*, A is an alphabet called the *input alphabet*, and δ is a function from $S \times A$ to S called the *state transition function*. The elements of S and A are called the *states* and the *input letters* of the automaton $\mathscr{A}$, respectively. The words over A are called the *input words* of $\mathscr{A}$.

Let $\mathscr{A} = (S, A, \delta)$ be an automaton. For any state s and any input letter a of $\mathscr{A}$, the image $\delta(s,a)$ of the ordered pair (s,a) under the state transition function δ is called the *action* of a on s, and simply denoted by sa, i.e.,

$$sa = \delta(s,a).$$

If $sa = t$, then it is said that the input letter a transitions the state s to the state t. The action of an input word w on a state s is inductively defined as below:

$$sw = \begin{cases} s & \text{if } w \text{ is the empty word } \epsilon, \\ (sa_1a_2\cdots a_{n-1})a_n & \text{if } w = a_1a_2\cdots a_n \text{ is a non-empty word.} \end{cases}$$

If $sw = t$, then it is said that the input word w transitions the state s to the state t. Clearly, for any state s and any input words w, u of $\mathscr{A}$, we have

$$(sw)u = s(wu).$$

The *state transition graph* of the automaton $\mathscr{A}$ is a labelled digraph having vertex set S and labelling set A that has an edge (s, a, t) from the state s to the state t with label a if and only if a is an input letter that transitions s to t. The *sketched state transition graph* of $\mathscr{A}$ is the digraph obtained by removing all loops from the state transition graph of $\mathscr{A}$.

2.4 Synchronising automata

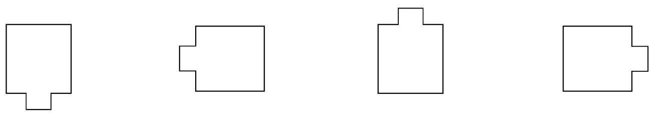
Let $\mathscr{A} = (S, A, \delta)$ be an automaton. As a generalisation of the action of an input word on a state, define the action Tw of an input word w of $\mathscr{A}$ on a set T of states of $\mathscr{A}$ as follows:

$$Tw = \{tw : t \in T\}.$$

The word w is called a *synchronising word* of T if w transitions all states in T to the same state. The sets of states having synchronising words are called *synchronisable sets* (Salomaa, 2012). The automaton $\mathscr{A}$ is said to be *synchronising* if its state set S is synchronisable. In this case, the synchronising words of S is called the *synchronising words* of $\mathscr{A}$. The following simplified application example of synchronising automata is introduced by Volkov (2009).

Example 2.1 (Volkov, 2009): Suppose that a certain device has a polygonal part. Such parts arrive at manufacturing sites in boxes and they need to be sorted and oriented before assembly.

Figure 1 Four possible orientations called 0, 1, 2, 3, respectively

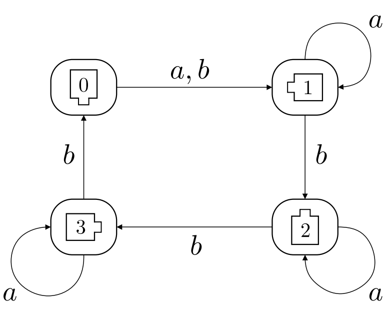


For simplicity, assume that only four initial orientations of the part shown in Figure 1 are possible. Further, suppose that prior the assembly the part should take the "bump-left" orientation. Thus one has to construct an orienter that will put the part in the prescribed position independently of its initial orientation. This goal can be achieved in the following sensor-free scheme. We put parts to be oriented on a conveyer belt which takes them to the assembly point (assume that the belt is moving from left to right), and let the stream of the parts encounter a series of obstacles a and b placed along the belt in the order *abbbabbba*. A obstacle b should be high enough in order that any part on the belt encounters this obstacle by its rightmost low angle. Being curried by the belt, the part then is forced to turn 90° clockwise. A obstacle a has the same effect whenever the part is in the orientation 0; otherwise it does not touch the part which therefore passes by without changing the orientation. The scheme can be described as an automaton $\mathscr{C}_4 = (S, A, \delta)$ in which $S = \{0, 1, 3, 2\}$, $A = \{a, b\}$, and the state transition function δ is defined as below:

$$\delta(0, b) = 1, \delta(1, b) = 2, \delta(2, b) = 3, \delta(3, b) = 0,$$
$$\delta(0, a) = 1, \delta(1, a) = 1, \delta(2, a) = 2, \delta(3, a) = 3.$$

The state transition graph of the automaton $\mathscr{C}_4$ is given as in Figure 2.

Figure 2 The actions of the obstacles



It is a routine matter to check that, under the actions of the obstacles, the parts can always be oriented to the orientation 1 when they are taken to the assembly. In other words, the input word ab^3ab^3a is a synchronising word of the automaton $\mathscr{C}_4$ that transitions all states to the state 1. The synchronising automaton $\mathscr{C}_4$ was originally invented by Černý (1964), and it is currently known as the 4*-state Černý automaton*.

2.5 The rank and the terminal words of an automaton

Let $\mathscr{A} = (S, A, \delta)$ be an automaton. Define the *anti-action* Tw^{-1} of an input word w of $\mathscr{A}$ on a set T of states of $\mathscr{A}$ as below:

$$Tw^{-1} = \{s \in S : sw \in T\}.$$

For an arbitrary state t of $\mathscr{A}$, we write tw^{-1} rather than $\{t\}w^{-1}$. Of course, the set tw^{-1} is non-empty if and only if $t \in Sw$. Moreover, when it is non-empty, the set tw^{-1} is *maximal* in the sets of states having synchronising word w. Let

$$S/w = \left\{tw^{-1} : t \in Sw\right\}. \tag{1}$$

Then S/w is a partition of the state set S such that two states are contained in the same class precisely when w transitions them to the same state. Evidently, the number of the classes in the partition S/w of S coincides with the cardinality of the set Sw. The rank $\mathbf{r}(w)$ of the input word w is defined by the cardinality of the set Sw, i.e,.

$$\mathbf{r}(w) = |Sw| = |S/w|.$$

An input word w of the automaton $\mathscr{A}$ is said to be *r-rank* if $\mathbf{r}(w) \leq r$, and it is called *exactly r-rank* if $\mathbf{r}(w) = r$.

The input words of $\mathscr{A}$ having the minimum rank are called the *terminal words*, while the rank of the terminal words is called the rank of $\mathscr{A}$, in notation $\mathbf{r}(\mathscr{A})$. Then

$$\mathbf{r}(\mathscr{A}) = \min_{w \in A^*} \mathbf{r}(w).$$

Clearly, an automaton is synchronising if and only if it is 1-rank, and the terminal words of a synchronising automaton are exactly the synchronising words.

3 The definition and decision of matching automata

This section is begun with defining matching automata. After some important properties of matching automata are revealed, a decision algorithm for matching automata is established.

Let $\mathscr{A} = (S, A, \delta)$ be an automaton. According to the definition of synchronisable sets, every one-element subset of the state set of $\mathscr{A}$ is synchronisable. To avoid the unnecessary discussions, the synchronisable sets of $\mathscr{A}$ containing at least two states are specially called *mergable sets*. Correspondingly, the synchronising words of a mergable set of $\mathscr{A}$ are specially called the *merging words*. Then the states in a mergable set can be *merged* to a single state under the action of a merging word. An input letter a of the automaton $\mathscr{A}$ is called a *defective letter* if its anti-action ta^{-1} on some state t contains at least two states (He et al., 2021). If this is the case, the set ta^{-1} is called a *defected set* of $\mathscr{A}$. The following lemma gives an elementary description for the binary mergable sets of automata.

Lemma 3.1: A binary set ε of states of an automaton $\mathscr{A}$ is mergable if and only if there exists an input word w of $\mathscr{A}$ such that εw is a binary set included in a defective set of $\mathscr{A}$. Moreover, if this is the case, the concatenation wa of the input word w and some defective letter a of $\mathscr{A}$ is a shortest merging word of ε, and εw is a binary mergable set included in a defective set of the form ta^{-1}.

Proof: Let ε be a binary set of states of $\mathscr{A}$. If w is an input word of $\mathscr{A}$ such that εw is a binary set included in a defective set ta^{-1}, then $\varepsilon(wa) = (\varepsilon w)a = t$, and thus ε is mergable. Conversely, if ε is mergable, then it has a shortest word say $a_1a_2\cdots a_n$. Let $w = a_1a_2\cdots a_{n-1}$ and $t = \varepsilon a_1a_2\cdots a_n$. It is certain that εw is a binary set since, otherwise, w is a merging word of ε shorter than $a_1a_2\cdots a_n$. Moreover, it follows by the equality $(\varepsilon w)a_n = \varepsilon(wa_n) = t$ that εw is a binary mergable set included in the defective set ta_{n-1}^{-1}. This completes the proof. ■

The *merging graph* of an automaton $\mathscr{A} = (S, A, \delta)$ is defined by the simple undirected graph $\mathbf{G}_{\mathscr{A}} = (S, E_{\mathscr{A}})$ in which $E_{\mathscr{A}}$ is the family of all binary mergable sets of $\mathscr{A}$. The automaton $\mathscr{A}$ is called a *matching automaton* if $\mathbf{G}_{\mathscr{A}}$ is a 1-regular graph or, alternatively, if $E_{\mathscr{A}}$ is a matching of the state set S. If $\mathscr{A}$ is matching, it is evident that $|E_{\mathscr{A}}| = |S|/2$; furthermore, Lemma 3.2 shows that the anti-action of an input word on an edge is also an edge, and the action of an input word on an edge is either an edge or a single state.

Lemma 3.2: Let $\mathscr{A} = (S, A, \delta)$ be a matching automaton with $E_{\mathscr{A}} = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k\}$, and w an input word of $\mathscr{A}$. Then there exists a permutation σ on the set $\mathbf{k} = \{1, 2, \ldots, k\}$ such that

$$\varepsilon_i w^{-1} = \varepsilon_{\sigma(i)}, \qquad \varepsilon_i w \subseteq \varepsilon_{\sigma^{-1}(i)} \qquad (i \in \mathbf{k}).$$

Proof: Under the hypothesis, $E_{\mathscr{A}}$ is a matching of the state set S. Furthermore, it is easy to illustrate that the sets $\varepsilon_1 w^{-1}, \varepsilon_2 w^{-1}, \ldots, \varepsilon_k w^{-1}$ are pairwise disjoint such that

$$\bigcup_{i\in\mathbf{k}} \varepsilon_i w^{-1} = \left(\bigcup_{i\in\mathbf{k}} \varepsilon_i\right) w^{-1} = S w^{-1} = S, \tag{2}$$

and thus

$$\sum_{i\in\mathbf{k}} \left|\varepsilon_i w^{-1}\right| = |S| = 2k. \tag{3}$$

Take an arbitrary edge ε from $E_{\mathscr{A}}$. Then every binary subset of the set εw^{-1} has to be an edge since, for any merging word u of ε, we have

$$\left|\left(\varepsilon w^{-1}\right) wu\right| \le |\varepsilon u| = 1.$$

In fact, the set εw^{-1} is the only binary subset of itself. Otherwise, it contains at least three pairwise distinct states say s, t, t'. This induces the contradiction that the matching $E_{\mathscr{A}}$ of the state set S contains both of the binary sets $\{s, t\}$ and $\{s, t'\}$. Now, it is reasonable to claim that $|\varepsilon w^{-1}| \le 2$ for any $\varepsilon \in E_{\mathscr{A}}$, and therefore

$$\sum_{i=0}^{k} \left|\varepsilon_i w^{-1}\right| \le 2k. \tag{4}$$

Applying equality (3) and inequality (4), we can get the following equalities:

$$\left|\varepsilon_1 w^{-1}\right| = \left|\varepsilon_2 w^{-1}\right| = \ldots = \left|\varepsilon_k w^{-1}\right| = 2.$$

This means that

$$E_{\mathscr{A}} = \left\{\varepsilon_1 w^{-1}, \varepsilon_2 w^{-1}, \ldots, \varepsilon_k w^{-1}\right\},$$

and hence there exists a permutation σ on the set $\mathbf{k}$ such that the equality $\varepsilon_i w^{-1} = \varepsilon_{\sigma(i)}$ holds for each $i \in \mathbf{k}$. Furthermore, $\varepsilon_i w \subseteq \varepsilon_{\sigma^{-1}(i)}$ follows since $\varepsilon_i = \varepsilon_{\sigma^{-1}(i)} w^{-1}$. ■

For a family Ω of subsets of the state set of an automaton $\mathscr{A} = (S, A, \delta)$, define the action Ωw and the anti-action Ωw^{-1} of an input word w of $\mathscr{A}$ on Ω as below:

$$\Omega w = \{\varepsilon w : \varepsilon \in \Omega\}, \qquad \Omega w^{-1} = \{\varepsilon w^{-1} : \varepsilon \in \Omega\}.$$

More general, define the action ΩL and the anti-action ΩL^{-1} of a set L of input words of $\mathscr{A}$ on Ω as below:

$$\Omega L = \bigcup_{w \in L} \Omega w, \qquad \Omega L^{-1} = \bigcup_{w \in L} \Omega w^{-1}.$$

The following lemma gives a characterisation for matching automata.

Lemma 3.3: Let $\mathscr{A} = (S, A, \delta)$ be an automaton, and Ω the family of all defected sets of $\mathscr{A}$. Then $\mathscr{A}$ is a matching automaton if and only if $\Omega\left(A^*\right)^{-1}$ is a matching of S. Moreover, if this is the case, $E_{\mathscr{A}} = \Omega\left(A^*\right)^{-1}$.

Proof: Let $\mathscr{A} = (S, A, \delta)$ be an automaton, and Ω the family of all defected sets of $\mathscr{A}$. Then

$$\Omega = \left\{ta^{-1} : t \in S,\ a \in A \text{ and } |ta^{-1}| \geq 2\right\}. \tag{5}$$

Moreover, it is evident that

$$\Omega = \Omega\epsilon^{-1} \subseteq \Omega\left(A^*\right)^{-1}. \tag{6}$$

Suppose that $\Omega\left(A^*\right)^{-1}$ is a matching of S. Then the sets in $\Omega\left(A^*\right)^{-1}$ are binary, so that the sets in Ω are binary. If ε is a set in $\Omega\left(A^*\right)^{-1}$, then $\varepsilon = (ta^{-1})w^{-1}$ for some $t \in S$, $a \in A$ and $w \in A^*$, so that $\varepsilon(wa) = t$, and hence $\varepsilon \in E_{\mathscr{A}}$. This means that

$$\Omega\left(A^*\right)^{-1} \subseteq E_{\mathscr{A}}. \tag{7}$$

If $\varepsilon \in E_{\mathscr{A}}$, then it follows by Lemma 3.1 that there exists an input word w of $\mathscr{A}$ such that εw is a binary set included in some set ta^{-1} in Ω. Since the sets in Ω are binary, this implies that $\varepsilon w = ta^{-1} \in \Omega$, so that $(\varepsilon w)w^{-1} \in \Omega\left(A^*\right)^{-1}$. Furthermore, since $\varepsilon \subseteq (\varepsilon w)w^{-1}$ and the sets in $\Omega\left(A^*\right)^{-1}$ are binary, we have $(\varepsilon w)w^{-1} = \varepsilon$, so that $\varepsilon \in \Omega\left(A^*\right)^{-1}$. This means that

$$E_{\mathscr{A}} \subseteq \Omega\left(A^*\right)^{-1}, \tag{8}$$

so that $E_{\mathscr{A}} = \Omega\left(A^*\right)^{-1}$, and hence $\mathscr{A}$ is a matching automaton.

For the converse, suppose that $\mathscr{A}$ is a matching automaton. Since every edge in $E_{\mathscr{A}}$ is a binary mergable set, we can see by Lemma 3.1 that the family Ω is non-empty. If a set ε in Ω contains three distinct states say s, t, p, then both of $\{s, t\}$ and $\{s, p\}$ are edges in $E_{\mathscr{A}}$. This contradicts to the assumption that $\mathscr{A}$ is a matching automaton, and thus the sets in Ω are all binary. Consequently, the family Ω is a subset of $E_{\mathscr{A}}$, and then it follows by Lemma 3.2 that formula (7) holds. Take an arbitrary edge ε from $E_{\mathscr{A}}$. Since the sets in Ω are binary, we can see by Lemma 3.1 that there exists an input word w such that $\varepsilon w \in \Omega$, so that $(\varepsilon w)w^{-1} \in \Omega\left(A^*\right)^{-1}$. Furthermore, applying formula (7) and Lemma 3.2, we can easily illustrate that $\varepsilon = (\varepsilon w)w^{-1}$. This means that formula (8) holds also, so that $E_{\mathscr{A}} = \Omega\left(A^*\right)^{-1}$, and hence the family $\Omega\left(A^*\right)^{-1}$ is a matching of S. ■

By the above lemma, it is enough to decide the matchingness of an automaton $\mathscr{A} = (S, A, \delta)$ by two steps: the first step is to compute the family $\Omega\left(A^*\right)^{-1}$ of subsets of S, the second one is to check whether the family $\Omega\left(A^*\right)^{-1}$ is a matching of S. In fact, the above two steps can be combined. The details are given in the following theorem.

Theorem 3.4: Algorithm 1 can decide the matchingness of an n-state automaton $\mathscr{A} = (S, A, \delta)$ over m input letters in $O(nm)$ times.

Proof: Let Ω be the family of all defected sets of $\mathscr{A}$. Put $X_0 = Z_0 = \Omega$ and define

$$X_i = \bigcup_{w \in A^*, \ell(w) \leq i} \Omega w^{-1},$$
$$Z_i = X_i - X_{i-1} \quad (i = 1, 2, 3, \ldots).$$

Then it is certain that

$$\Omega\left(A^*\right)^{-1} = \bigcup_{i \geq 0} X_i. \tag{9}$$

Moreover, it is routine to check that

$$\begin{aligned} \bigcup_{j=0}^{i} Z_j &= X_i \subseteq X_{i+1} \\ &= X_0 \bigcup \left(X_i A^{-1}\right) \quad (i = 0, 1, 2, \ldots). \end{aligned} \tag{10}$$

so that

$$Z_j A^{-1} \subseteq X_j A^{-1} \subseteq X_{j+1} \subseteq X_i \qquad (0 \leq j < i),$$

and hence

$$\begin{aligned} Z_i &= \left[X_0 \bigcup \left(X_{i-1}A^{-1}\right)\right] - X_{i-1} \\ &= \bigcup_{j=0}^{i-1} Z_j A^{-1} - X_{i-1} \\ &= Z_{i-1}A^{-1} - X_{i-1} \quad (i = 1, 2, \ldots). \end{aligned}$$

For any non-negative integer i, it is certain that $X_i = X_{i+1}$ if and only if $Z_{i+1} = \emptyset$. If this is the case, we can easily see by formula (10) that

$$X_i = X_{i+1} = \ldots, \qquad \emptyset = Z_i = Z_{i+1} = \ldots,$$

And then $X_i = \Omega\left(A^*\right)^{-1}$ follows equality (9). Therefore, in the case that $\Omega \neq \emptyset$, there exists the minimum non-negative integer say k such that

$$Z_k \neq \emptyset, \qquad Z_{k+1} = Z_{k+2} = \ldots = \emptyset,$$
$$X_0 \subsetneq X_1 \subsetneq \ldots \subsetneq X_k = X_{k+1} = \ldots = \Omega\left(A^*\right)^{-1}.$$

By Lemma 3.3, the automaton $\mathscr{A}$ is matching if and only if the following statements are true:

1 n is an even number

2 the family Z_0 is non-empty and its members are pairwise disjoint binary sets

3 for each positive integer i, if $Z_i \neq \emptyset$, then the members of Z_i are binary sets which are not only pairwise disjoint but also disjoint to the members of X_{i-1}

4 if k is the minimum integer such that $Z_k = \emptyset$, then $|X_k| = n/2$.

Algorithm 1 Deciding the matchingness of an automaton

```
Input: an $n$-state automaton $\mathscr{A} = (S, A, \delta)$ over $m$ input letters
Output: a string W and a set $X$
Initialise W = NO, $X = \{\}$.
1: If $n$ is odd, return; otherwise, continue.
2: Initialise $Y = S$ and $Z = \{\}$.
3: Consider the sets of the form $sa^{-1}$ $(s \in Y, a \in A)$ in the following rules [(a)–(d)]:
    (a) if $|sa^{-1}| = 1$, skip;
    (b) if $sa^{-1}$ is a binary set whose members are unlabelled, then label its members for each other, and add $sa^{-1}$ into $Z$;
    (c) if $sa^{-1}$ is a binary set whose members are labelled for each other, skip;
    (d) otherwise, return.
4: If $Z = \{\}$, return; otherwise, continue.
5: Put $X = X \bigcup Z$, $Y = Z$, $Z = \{\}$.
6: Consider the sets of the form $\varepsilon a^{-1}$ $(\varepsilon \in Y, a \in A)$ in the following rules [(e)–(g)]:
    (e) if $\varepsilon a^{-1}$ is a binary set whose members are unlabelled, then label its members for each other, and add $\varepsilon a^{-1}$ into $Z$;
    (f) if $\varepsilon a^{-1}$ is a binary set whose members are labelled for each other, skip;
    (g) otherwise, Return.
7: Consider the sets $Z$ and $X$ in the following rules [(h)–(j)]:
    (h) if $Z \neq \{\}$ and $|X| + |Z| < \hat{n}$, turn to the step 5;
    (i) if $Z = \{\}$ and $|X| \neq \hat{n}$, return;
    (j) if $Z = \{\}$ and $|X| = \hat{n}$, let W=YES, return.
```

Beginning with Z_0 and ending with the first integer say m such that $Z_k = \emptyset$, Algorithm 1 computes the pairs $(X_0, Z_0), (X_1, Z_1), \ldots$ one by one, and decides the matchingness of $\mathscr{A}$ in rules 1–4. In the process, to avoid the troublesome comparisons, the members of every new member in the families Z_i's are labelled for each other. Steps 3 and 4 of Algorithm 1 are aimed to check statement 2 which take $O(nm)$ times. Steps 5–7 are designed to check statements 3 and 4. Since $\sum |Z_i| \leq n/2$, these steps take times at most $O(nm)$. Therefore the time complexity of Algorithm 1 is $O(nm)$. ■

4 The r-rank input words of matching automata

For any non-negative integers n and r with $r \leq n$, let

$$M(n, r) = \sum_{i=0}^{n-r} i.$$

The following theorem is the main result of this section. It illustrates that Pin conjecture (and hence rank conjecture) is true for matching automata since

$$M(n, r) = \frac{(n-r)^2 + (n-r)}{2} \leq (n-r)^2.$$

Theorem 4.1: Let $\mathscr{A} = (S, A, \delta)$ be an n-state matching automaton, and r a non-negative integer not exceeding n. Then $\mathscr{A}$ is of rank $n/2$, and it has an r-rank input word if and only if $r \geq n/2$. Moreover, if $\mathscr{A}$ has an r-rank input word, then it has an r-rank input word of length not exceeding $M(n, r)$.

Proof: Under the hypothesis, the edge set $E_{\mathscr{A}}$ of the merging graph $\mathbf{G}_{\mathscr{A}}$ of $\mathscr{A}$ is a matching of the state set S of $\mathscr{A}$, and hence it consists of $n/2$ edges. Suppose that

$$E_{\mathscr{A}} = \{\{s_i, t_i\} : i = 1, 2, \ldots, n/2\}.$$

For any terminal word w of $\mathscr{A}$, it follows by Lemma 3.2 that

$$\mathbf{r}(w) = |Sw| \geq |\{s_i w : i = 1, 2, \ldots, n/2\}| = n/2.$$

If $\mathbf{r}(w) > n/2$, then the set Sw includes some edge ε_i in $E_{\mathscr{A}}$, so that $|Sw| > |Sww_i|$. This induces a contradiction since $\mathbf{r}(w) = \min_{u \in A^*} \mathbf{r}(u)$. Therefore $\mathbf{r}(\mathscr{A}) = \mathbf{r}(w) = n/2$, and hence $\mathscr{A}$ has an r-rank input word if and only if $r \geq n/2$.

Take a shortest merging word w_ε for each edge $\varepsilon \in E_{\mathscr{A}}$, and let

$$m = \max\{\ell(w_\varepsilon) : \varepsilon \in E_{\mathscr{A}}\}.$$

Then $m = \ell(w_\varrho)$ for some $\varrho \in E_{\mathscr{A}}$. For any positive integer k less than m, suppose that $w_\varrho = uv$ in which u is the left factor of w_ϱ having length $m - k$. It follows by Lemma 3.2 that $\varrho u = \varepsilon$ for some $\varepsilon \in E_{\mathscr{A}}$. Moreover, it is easy to check that the words uw_ε and v are merging words of the edges ϱ and ε, respectively, so that

$$\ell(w_\varrho) \leq \ell(uw_\varepsilon) = \ell(u) + \ell(w_\varepsilon),$$
$$\ell(w_\varepsilon) \leq \ell(v) = \ell(w_\varrho) - \ell(u),$$

and hence

$$\ell(w_\varepsilon) = \ell(w_\varrho) - \ell(u) = k.$$

Consequently, we have

$$\{\ell(w_\varepsilon) : \varepsilon \in E_{\mathscr{A}}\} = \{1, 2, \ldots, m\} \subseteq \{1, 2, \ldots, n/2\}. \tag{11}$$

Figure 3 The sketched transition graph of the automaton $\mathscr{M}_n$

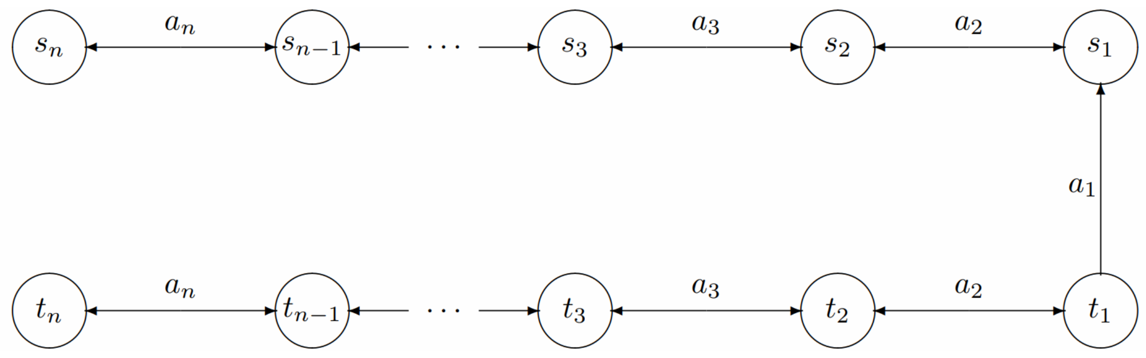
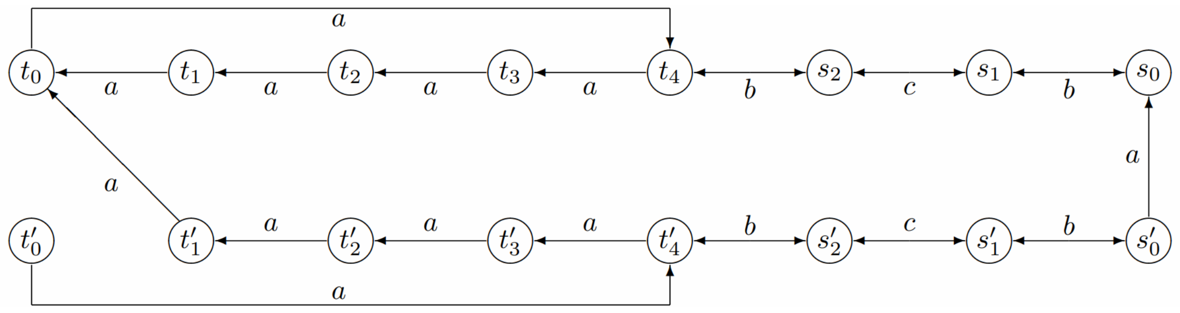


Figure 4 The sketched transition graph of the automaton $\mathscr{A}$



Put $X_0 = E_{\mathscr{A}}$ and $u_0 = \epsilon$. For each positive integer i, unless $X_{i-1} = \emptyset$, take an edge ε_i from X_{i-1} in the rule that

$$\ell\left(w_{\varepsilon_i}\right) = \min_{\varepsilon \in X_{i-1}} \ell(w_\varepsilon),$$

and define

$$X_i = X_0 \bigcap X_{i-1} w_{\varepsilon_i}, \qquad u_i = w_{\varepsilon_1} w_{\varepsilon_2} \dots w_{\varepsilon_i}.$$

Then $X_d = \emptyset$ for some positive integer d not exceed $n/2$ since it is evident that

$$|X_0| > |X_1| > |X_2| > \dots. \tag{12}$$

For each $i = 0, 1, \dots, d$, it is a routine matter to illustrate by Lemma 3.2 that u_i is a merging word of an edge ε if and only if $\varepsilon u_i \notin X_i$, so that

$$\mathbf{r}(u_i) = |S u_i| = n/2 + |X_i|. \tag{13}$$

Clearly, u_0 is an n-rank input word of $\mathscr{A}$ with length $0 = M(n, n)$. Since $\mathscr{A}$ has at least one edge that is a defected set, the word u_1 is certainly a defective letter of $\mathscr{A}$, and hence it is an $(n-1)$-rank input word of $\mathscr{A}$ with length $1 = M(n, n-1)$. Inductively assume that u_i is a $(r+1)$-rank input word of $\mathscr{A}$ having length not exceed $M(n, r+1)$. In the case that $\mathbf{r}\left(u_i\right) < r+1$, it is certain that u_i is also a r-rank input word of $\mathscr{A}$ having length

$$\ell\left(u_i\right) \le M(n, r+1) < M(n, r).$$

In the case that $\mathbf{r}\left(u_i\right) = r+1$, it follows by inequality (12) and equality (13) that u_{i+1} is an r-input word of $\mathscr{A}$. Moreover, by equality (13), we can see that

$$|X_i| = \mathbf{r}\left(u_i\right) - k = r + 1 - k.$$

This implies by formula (11) that

$$\ell\left(w_{\varepsilon_{i+1}}\right) = \min_{\varepsilon \in X_i} \ell(w_\varepsilon) \le k - (r+1-k) + 1 = n - r,$$

so that

$$\begin{aligned}\ell\left(u_{i+1}\right) &= \ell\left(u_1\right) + \ell\left(w_{\varepsilon_{i+1}}\right)\\ &\le M(n, r+1) + (n-r) = M(n, r).\end{aligned}$$

This completes the proof. ■

Remark 4.2: Algorithm 1 can be extended as follows:

1 if ε is a binary defected set of $\mathscr{A}$ obtained as the set sa^{-1} for some $s \in S$ and $a \in A$, then associate ε with the letter $w_\varepsilon = a$ and the integer $i_\varepsilon = 1$

2 if ε is an edge of $E_{\mathscr{A}}$ obtained as the set $\varepsilon' a^{-1}$ for some $\varepsilon' \in Y$ and $a \in A$, then associate ε with the word $w_\varepsilon = w_{\varepsilon'} a$ and the integer $i_\varepsilon = i_{\varepsilon'} + 1$.

When $\mathscr{A}$ is matching, for any $\varepsilon \in E_{\mathscr{A}}$, it is evident that w_ε is a shortest merging words of ε with $\ell(w_\varepsilon) = i_\varepsilon$. Following from formula (11), we can see that

$$\sum_{\varepsilon \in E_{\mathscr{A}}} i_\varepsilon \le \sum_{i=1}^{n/2} i = \frac{n^2 + 2n}{8}.$$

Therefore, the extend Algorithm 1 is of time complexity $O\left(nm+n^2\right)$.

Corollary 4.3: Given an n-state automaton $\mathscr{A}=(S,A,\delta)$ over m input letters. When $\mathscr{A}$ is matching, Algorithm 2 is available to compute an r-rank input word of $\mathscr{A}$ in $O(nm+n^2)$ times such that the found word has length at most $M(n,r)$.

Algorithm 2 Computing an r-rank input word of a matching automaton

```
Input: an $n$-state automaton $\mathscr{A}=(S,A,\delta)$ over $m$ input letters and an integer $r$.
Output: a special symbol or an $r$-rank input word $u$ of $\mathscr{A}$.
Initialise $u=\triangle$.
1: perform the extended Algorithm 1, if $\mathscr{A}$ is not matching, return; otherwise, continue.
2: if $r<n/2$, return; otherwise, continue.
3: let $X=E_{\mathscr{A}}$ and $u=\epsilon$.
4: find the edge $\varepsilon$ such that $i_\varepsilon=\min\{i_{\varepsilon'}:\varepsilon'\in X\}$ and put $u=uw_\varepsilon$.
5: compute the set $X=\{\varepsilon' w_\varepsilon:\varepsilon'\in X$ and $|\varepsilon' w_\varepsilon|=2\}$.
6: if $|X|>n-r$, turn to the step 4; otherwise, return.
```

Proof: It follows from the proof of Theorem 4.1 that Algorithm 2 is available to compute an r-rank input word of $\mathscr{A}$ which has length at most $M(n,r)$. As pointed out in Remark 4.2, the step 2 may takes $O\left(nm+n^2\right)$ times. Since $\mathscr{A}$ has $n/2$ edges, and the number of the edges in X is strictly decreasing, to perform the step 4 in all possible loops needs $O\left(n^2\right)$ times. Furthermore, since the output word u is of length $O\left(n^2\right)$, to perform the step 5 in all possible loops also needs $O\left(n^2\right)$ times. Hence the time complexity of Algorithm 2 is $O(nm+n^2)$.

Remark 4.4: Essentially, Algorithm 2 was invented by Pin (1983). In general, the r-rank input word of a matching automaton obtained by performing Algorithm 2 is not the shortest one. For example, let $\mathscr{A}_1$ be an automaton having the sketched transition graph given in Figure 3.

Then $\mathscr{A}_1$ has a unique defective letter a and two defected sets $\{s_0,s_0'\}$, $\{t_1,t_1'\}$. Performing the extended Algorithm 1, we can obtain that $E_{\mathscr{A}}=\{\varepsilon_i : i=1,2,\ldots,8\}$ where

$$\varepsilon_1=\{t_1,t_1'\},\varepsilon_2=\{s_0,s_0'\},\varepsilon_3=\{t_2,t_2'\},\varepsilon_4=\{s_1,s_1'\},$$
$$\varepsilon_5=\{t_3,t_3'\},\varepsilon_6=\{s_2,s_2'\},\varepsilon_7=\{t_4,t_4'\},\varepsilon_8=\{t_0,t_0'\},$$

and thus $\mathscr{A}_1$ is matching. The shortest synchronising word $w_i\,(i=1,2,\ldots,8)$ of the edge ε_i obtained in this process is given as below:

$$w_1=w_2=a,\quad w_3=a^2,\quad w_4=ba,\quad w_5=a^3,$$
$$w_6=cba,\quad w_7=a^4,\quad w_8=a^5.$$

Continuing to perform the left steps of Algorithm 2, we may obtain a terminal input word $a^5(ba)a^3$ of $\mathscr{A}_1$ which has length 10. Direct calculation shows that $abacba^3$ is a terminal input word of $\mathscr{A}_1$ which has length only 8.

5 The extreme matching automata

In this section, we consider the so-called n-state *extreme matching automaton* $\mathscr{M}_n=(M,A,\delta)$ having the sketched transition graph given in Figure 4.

Clearly, $\mathscr{M}_n$ has a unique defective letter a and a unique defected set $\{s_0,t_0\}$. Let

$$\varepsilon_i=\{s_i,t_i\},\quad u_i=a_ia_{i-1}\ldots a_1 \qquad (i=1,2,\ldots,k).$$

It is easy to see that $\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k$ are all edges of the merging graph $\mathbf{G}_{\mathscr{M}_n}$ of $\mathscr{M}_n$, and that the words $u_1,u_2,\ldots,u_k$ are the unique shortest synchronising words of the edges $\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k$, respectively. Therefore, $\mathscr{M}_n$ is a matching automaton with

$$E_{\mathscr{M}_n}=\{\varepsilon_1,\varepsilon_2,\ldots,\varepsilon_k\}.$$

Two input letters a_i,a_j of $\mathscr{M}_n$ are said to be *commutative* if $sa_ia_j=sa_ja_i$ for all $s\in M$. For any input words w,w' of $\mathscr{M}_n$, we write $w\sim w'$ if $w=ua_ia_jv$ and $w'=ua_ja_iv$ for two commutative letters a_i,a_j and two input words u,v of $\mathscr{M}_n$. More general, we write $w\overset{*}{\sim}w'$ if there is a sequence $w=w_0,w_1,\ldots,w_m=w'$ of input words of $\mathscr{M}_n$ such that $w_i\sim w_{i+1}$ for each $i=0,1,\ldots,m-1$. If $w\overset{*}{\sim}w'$, then we call w' a *swapped word* of w. Recall that an equivalence ρ on A^* is called a *congruence* if $w\,\rho\,w'$ implies $uwv\,\rho\,uw'v$ for any $u,v\in A^*$. The following lemma is evident.

Lemma 5.1: Two input letters a_i,a_j of $\mathscr{M}_n$ are commutative if and only if $Ta_ia_j=Ta_ja_i$ for any subset T of M, and if and only if $i-j\neq\pm1$. The relation $\overset{*}{\sim}$ on A^* is a congruence. Moreover, if $w\overset{*}{\sim}w'$, then $Mw=Mw'$ and $\mathbf{r}(w)=\mathbf{r}(w')$. ■

It is evident that the empty word ϵ is the unique shortest n-rank input word of $\mathscr{M}_n$. In what follows, we consider the shortest r-rank input words of $\mathscr{M}_n$ where r is an arbitrary integer in the interval $[n/2,n-1]$. If $T=\{s_i,t_i,\ldots\}$ is a subset of M including an edge ε_i, for the sake of convenience, we formally write T as $\{\varepsilon_i,\ldots\}$. Some fundamental properties of the shortest r-rank input words of $\mathscr{M}_n$ are given as below:

Lemma 5.2: Let $w=x_1x_2\ldots x_m\,(x_1,x_2,\ldots,x_m\in A)$ be a shortest r-rank input word of $\mathscr{M}_n$. Then the following statements are true for any meaningful integers i and j:

1 $|Mx_1x_2\ldots x_i|>|Mx_1x_2\ldots x_ix_{i+1}|$ if and only if $x_{i+1}=a_1$ and $\varepsilon_1\subseteq Mx_1x_2\ldots x_i$

2 $Mx_1x_2\ldots x_i=Mx_1x_2\ldots x_j$ if and only if $i=j$.

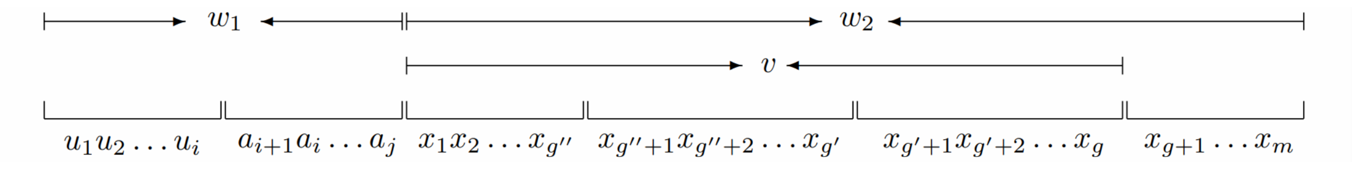
Proof: Since ε_1 is the unique defected set of $\mathscr{M}_n$, and a_1 is the unique letter acting as a merging word of ε_1, statement 1 holds. If $Mx_1x_2\ldots x_i=Mx_1x_2\ldots x_j$ for some integers i,j with $1\le i<j\le m$, then

$$\begin{aligned}Mw&=M(x_1x_2\ldots x_j)(x_{j+1}\ldots x_m)\\&=M(x_1x_2\ldots x_i)(x_{j+1}\ldots x_m).\end{aligned}$$

Table 1 All possible cases of the sets $T_2, T_2a_ia_{i+1}a_i, T_2a_ia_{i+1}$ and $T_2a_{i+1}a_i$

Case	T_2	$T_2a_ia_{i+1}a_i$	$T_2a_ia_{i+1}$	$T_2a_{i+1}a_i$
1	$\{s_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{s_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$
2	$\{\varepsilon_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{s_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$
3	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$	$\{s_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{s_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{s_{i-1}, s_i, s_{i+1}\}$
4	$\{s_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{\varepsilon_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, \varepsilon_{i+1}\}$
5	$\{s_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$	$\{s_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, s_{i+1}\}$
6	$\{\varepsilon_{i-1}, \varepsilon_i, s_{i+1}\}$	$\{s_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, s_i, \varepsilon_{i+1}\}$	$\{s_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$
7	$\{s_{i-1}, s_i, s_{i+1}\}$	$\{s_{i-1}, s_i, s_{i+1}\}$	$\{s_{i-1}, s_i, s_{i+1}\}$	$\{s_{i-1}, s_i, s_{i+1}\}$
8	$\{\varepsilon_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$	$\{\varepsilon_{i-1}, \varepsilon_i, \varepsilon_{i+1}\}$

Figure 5 A decomposition of the r-rank input word w'' of $\mathscr{M}_n$



This leads to the contradiction that $(x_1x_2\dots x_i)$ $(x_{j+1}\dots x_m)$ is an r-rank input word of $\mathscr{M}_n$ shorter than w. Therefore statement 2 is also true. ■

Lemma 5.3: Let w be a shortest r-rank input word of $\mathscr{M}_n$, w' a swapped word of w, and v' a swapped word of some factor v of w. Then the following statements are true:

1 the word w' is also a shortest r-rank input word of $\mathscr{M}_n$

2 the word v' is a factor of some swapped word of w

3 the word v' has no factor of the form $a_ia_i\ (i = 1, 2, \dots, n/2)$

4 the word v' has no factor of the form $a_ia_{i+1}a_i\ (i = 2, 3, \dots, n/2 - 1)$.

Proof: The swapped word w' of w is also a shortest r-rank input word of $\mathscr{M}_n$ since it is evident that $\ell(w) = \ell(w')$ and it follows from Lemma 5.1 that $\mathbf{r}(w) = \mathbf{r}(w')$. Since v is a factor of w, there have to be input words u_1, u_2 of $\mathscr{M}_n$ such that $w = u_1vu_2$. Observing from Lemma 5.1 that the relation $\overset{*}{\sim}$ on A^* is a congruence, we have $w = u_1vu_2 \overset{*}{\sim} u_1v'u_2$, and hence v' is a factor of the swapped word $u_1v'u_2$ of w.

If there is an integer i with $1 \le i \le n/2$ such that $v' = v_1a_ia_iv_2$ for some input words v_1, v_2 of $\mathscr{M}_n$, then $u_1v'u_2 = u_1v_1a_ia_iv_2u_2$ which is a swapped word of w and hence a shortest r-input word of $\mathscr{A}$. It is routine to check that, for any $s \in M$,

$$sa_ia_i = \begin{cases} s_1 = sa_i & \text{if } s \in \varepsilon_1 \text{ and } i = 1, \\ s & \text{otherwise,} \end{cases}$$

so that

$$Mu_1v_1a_ia_i = \begin{cases} Mu_1v_1 & \text{if } \varepsilon_1 \not\subseteq Mu_1v_1, \\ Mu_1v_1a_i & \text{if } \varepsilon_1 \subseteq Mu_1v_1. \end{cases}$$

This contradicts to Lemma 5.2(2). Hence v' has no factor of the form a_ia_i.

Hypothesise that i is an integer with $1 < i < n/2$ such that $v' = v_1(a_ia_{i+1}a_i)v_2$ for some input words v_1, v_2 of $\mathscr{M}_n$, and let

$$T_1 = Mu_1v_1 - \left(\varepsilon_{i-1} \bigcup \varepsilon_i \bigcup \varepsilon_{i+1}\right),$$
$$T_2 = Mu_1v_1 \bigcap \left(\varepsilon_{i-1} \bigcup \varepsilon_i \bigcup \varepsilon_{i+1}\right).$$

Then it is evident that $T_1a_ia_{i+1}a_i = T_1$, and thus

$$Mu_1v_1(a_ia_{i+1}a_i) = T_1 \bigcup (T_2a_ia_{i+1}a_i).$$

Furthermore, for each $j = i-1, i, i+1$, it is clear that $t_j \in T_2$ if and only if $\varepsilon_j \subseteq T_2$. Table 1 exhibits all possible cases of the sets $T_2, T_2a_ia_{i+1}a_i, T_2a_ia_{i+1}$ and $T_2a_{i+1}a_i$.

Observing that $T_2a_ia_{i+1}a_i$ coincides with one of the sets T_2, $T_2a_ia_{i+1}$ and $T_2a_{i+1}a_i$, we claim that $Mu_1v_1(a_ia_{i+1}a_i)v_2u_2$ coincides with one of the sets $Mu_1v_1v_2u_2$, $Mu_1v_1a_ia_{i+1}v_2u_2$ and $Mu_1v_1a_{i+1}a_iv_2u_2$. Since $u_1v_1(a_ia_{i+1}a_i)v_2u_2 \overset{*}{\sim} w$ and hence $Mu_1v_1(a_ia_{i+1}a_i)v_2u_2 = Mw$, this contradicts to the assumption that w is a shortest k-rank input word of $\mathscr{M}_n$. Therefore v' has no factor of the form $a_ia_{i+1}a_i\ (1 < i < n/2)$. ■

The following theorem is the main result of this section:

Theorem 5.4: Given an integer r with $n/2 \le r < n$. A word in A^* is a shortest r-rank input word of $\mathscr{M}_n$ if and only if it is a swapped word of the word $u_1u_2\dots u_{n-r}$.

Proof: It is routine to check that, for any positive integer i with $i \le n/2$,

$$Mu_1u_2\dots u_i = \{s_1, s_2, \dots, s_i, \varepsilon_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_{n/2}\}, \tag{14}$$

so that

$$\mathbf{r}(u_1u_2\dots u_i) = n - i,$$

and hence

$$\mathbf{r}(u_1u_2\dots u_{n-r}) = r.$$

Accordingly to Lemma 5.1, Lemma 5.3 and the above equality, the desired result is equivalent to that, if w is a shortest r-rank input word of $\mathscr{M}_n$, then each one of the words $u_1u_2\dots u_i$'s $(i = 1, 2, \dots, n - r)$ is a left factor of some swapped word of w.

Let w be an arbitrary shortest r-rank input word of $\mathscr{M}_n$. Then it follows from Lemma 5.2 that the initial of w has to be a_1, and hence $u_1 = a_1$ is a left factor of w. Inductively hypothesise that $u_1u_2\dots u_i$ $(1 \le i < n - r)$ is a left factor of some swapped word w' of w. In what follows, we show that $u_1u_2\dots u_iu_{i+1}$ is also a left factor of some swapped word of w by giving an induction to the left factors of u_{i+1}.

At first, supposing that $w' = u_1u_2\dots u_iu$, we can see by equality (14) and Lemma 5.2(2) that the initial of u is a_{i+1}, so that $u_1u_2\dots u_ia_{i+1}$ is also a left factor of the swapped word w' of w. Next, we inductively hypothesise that some word $u_1u_2\dots u_ia_{i+1}a_i\dots a_j$ $(1 < j \le i + 1)$ is a left factor of a swapped word w'' of w, and suppose that

$$w'' = w_1w_2,$$

where

$$w_1 = u_1u_2\dots u_ia_{i+1}a_i\dots a_j,$$
$$w_2 = x_1x_2\dots x_m \qquad (x_1, x_2, \dots, x_m \in A).$$

To show that the word $u_1u_2\dots u_ia_{i+1}a_i\dots a_ja_{j-1}$ is also a left factor of some swapped word of w and then complete the proof, we need Lemmas 5.5–5.9.

Lemma 5.5: The letter a_{j-1} appears in the word w_2.

Proof: Otherwise, observing from equality (14) that

$$\begin{aligned} Mw_1 &= \{s_1, s_2, \dots, s_i, \varepsilon_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_m\}a_{i+1}a_i\dots a_j \\ &= \{s_1, s_2, \dots, s_{i-1}, \varepsilon_i, s_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_m\}a_i\dots a_j \\ &= \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ &= \{s_1, s_2, \dots, s_{j-2}, \varepsilon_{j-1}, s_j, \dots, s_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_m\}, \end{aligned} \tag{15}$$

we can routinely show that, for any left factor u of w_2,

$$\varepsilon_h \bigcap Mw_1u = s_h \qquad (h = 1, 2, \dots, j - 2).$$

By virtue of Lemma 5.2 (1) and Lemma 5.1, this leads to the contradiction that

$$\mathbf{r}(w'') = |Mw''| = |Mw_1| = n - i > r.$$

Therefore the letter a_{j-1} has to appear in the word w_2. ■

Hereinafter, we suppose that x_g is the first appearance of the letter a_{j-1} in w_2, and let

$$v = x_1x_2\dots x_g.$$

Then we have

Lemma 5.6: The letters $a_1, a_2, \dots, a_{j-1}$ do not appear in the word $x_1x_2\dots x_{g-1}$.

Proof: Of course, the letter a_{j-1} does not appear in $x_1x_2\dots x_{g-1}$. If some letter in $a_1, a_2, \dots, a_{j-2}$ appears in $x_1x_2\dots x_{g-1}$, suppose that $x_{g'}$ is the first one of such appearances. Then it is certain that the letters $a_1, a_2, \dots, a_{j-1}$ do not appear in the word $x_1x_2\dots x_{g'-1}$. This implies by Lemma 5.1 that $x_{g'}$ is commutative with the letters $x_1, x_2, \dots, x_{g'-1}$, so that

$$w_1(x_1x_2\dots x_{g'-1})x_{g'} \overset{*}{\sim} w_1x_{g'}(x_1x_2\dots x_{g'-1}).$$

Consequently, applying Lemma 5.1 and equality (15), we may obtain the following equality contradicting to Lemma 5.2(2):

$$\begin{aligned} & Mw_1(x_1x_2\dots x_{g'-1})x_{g'} \\ &= Mw_1x_{g'}(x_1x_2\dots x_{g'-1}) \\ &= \{s_1, s_2, \dots, s_{j-2}, \varepsilon_{j-1}, s_j, \dots, s_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_m\} \\ & x_{g'}x_1x_2\dots x_{g'-1} \\ &= \{s_1, s_2, \dots, s_{j-2}, \varepsilon_{j-1}, s_j, \dots, s_{i+1}, \varepsilon_{i+2}, \dots, \varepsilon_m\} \\ & x_1x_2\dots x_{g'-1} \\ &= Mw_1(x_1x_2\dots x_{g'-1}). \end{aligned}$$

Therefore, it is sure that the letters $a_1, a_2, \dots, a_{j-1}$ do not appear in $x_1x_2\dots x_{g-1}$. ■

Lemma 5.7: If $x_{g'}$ is the last appearance of the letter a_j in the word $x_1x_2\dots x_{g-1}$, then there is an appearance $x_{g''}$ of the letter a_{j+1} in $x_1x_2\dots x_{g'-1}$ such that the letters a_{j-1}, a_j, a_{j+1} do not appear in $x_{g''+1}x_{g''+2}\dots x_{g'-1}$.

Proof: Suppose that $x_{g'}$ is the last appearance of a_j in $x_1x_2\dots x_{g-1}$. If a_{j+1} does not appear in $x_1x_2\dots x_{g'-1}$, then it follows from Lemma 5.1 and Lemma 5.6 that the letter $x_{g'} = a_j$ commutes with those in $x_1x_2\dots x_{g'-1}$, so that

$$\begin{aligned} & w_1(x_1x_2\dots x_{g'-1})x_{g'} \\ &= (u_1u_2\dots u_i)(a_{i+1}a_i\dots a_{j+1}a_j)(x_1x_2\dots x_{g'-1})a_j \\ &\overset{*}{\sim} (u_1u_2\dots u_i)(a_{i+1}a_i\dots a_{j+1}a_j)a_j(x_1x_2\dots x_{g'-1}) \\ &= (u_1u_2\dots u_i)(a_{i+1}a_i\dots a_{j+1})(a_ja_j)(x_1x_2\dots x_{g'-1}), \end{aligned}$$

and thus the factor $w_1(x_1x_2\dots x_{g'-1})x_{g'}$ of w'' has a swapped word which has a factor a_j^2. This contradicts to Lemma 5.3(3), yielding that the letter a_{j+1} appears in $x_1x_2\dots x_{g'-1}$. Let $x_{g''}$ be the last appearance of a_{j+1} in $x_1x_2\dots x_{g'-1}$. Then the r-rank input word w'' of $\mathscr{M}_n$ can be diagrammatised as below:

Of course, the letters a_{j-1}, a_{j+1} do not appear in $x_{g''+1}x_{g''+2}\dots x_{g'-1}$. If the letter a_j appears in $x_{g''+1}x_{g''+2}\dots x_{g'-1}$, for the last one say $x_{g''+l}$ of such appearances, we have

$$\begin{aligned}&x_{g''+1}x_{g''+2}\dots x_{g'-1}x_{g'}\\&= x_{g''+1}x_{g''+2}\dots x_{g'-1}a_j\\&\overset{*}{\sim} x_{g''+1}x_{g''+2}\dots x_{g''+l-1}(a_j a_j)x_{g''+l+1}\dots x_{g'-1}.\end{aligned}$$

Contradicting to Lemma 5.3(3), this means that the factor $x_{g''+1}x_{g''+2}\dots x_{g'-1}x_{g'}$ of w'' has a factor $a_j a_j$. Hence a_j also does not appear in $x_{g''+1}x_{g''+2}\dots x_{g'-1}$. ■

Lemma 5.8: If the letter a_j appears in the word $x_1x_2\dots x_{g-1}$, then there exist an integer h with $j+1 \le h \le k$ such that

$$v \overset{*}{\sim} v_1\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2$$

for some words $v_1, v_2 \in A^*$ in which v_1 is either the empty word or a left factor of v terminated by some letter in $a_j, a_{j+1}, \dots, a_h$.

Proof: Suppose that $x_{g'}$ is the last appearance of a_j in $x_1x_2\dots x_{g-1}$. Then it follows from Lemma 5.7 that there is an appearance $x_{g''}$ of a_{j+1} in $x_1x_2\dots x_{g'-1}$ such that a_{j-1}, a_j, a_{j+1} do not appear between $x_{g''}$ and $x_{g'}$. This implies by Lemma 5.1 and Lemma 5.6 that

$$\begin{aligned}v &= (x_1x_2\dots x_{g''-1})a_{j+1}(x_{g''+1}x_{g''+2}\dots x_{g'-1})\\&\quad a_j(x_{g'+1}x_{g'+2}\dots x_{g-1})a_{j-1}\\&\overset{*}{\sim} (x_1x_2\dots x_{g''-1})a_{j+1}(x_{g''+1}x_{g''+2}\dots x_{g'-1})\\&\quad a_j a_{j-1}(x_{g'+1}x_{g'+2}\dots x_{g-1})\\&\overset{*}{\sim} (x_1x_2\dots x_{g''-1})a_{j+1}a_j a_{j-1}\\&\quad (x_{g''+1}x_{g''+2}\dots x_{g'-1})(x_{g'+1}x_{g'+2}\dots x_{g-1}).\end{aligned} \tag{16}$$

If there is an appearance $x_{g'''}$ of a_{j+2} in $x_1x_2\dots x_{g''-1}$ such that the letters $a_{j-1}, a_j, a_{j+1}, a_{j+2}$ do not appear between $x_{g'''}$ and $x_{g''}$, similar to formula (16), we may obtain that

$$\begin{aligned}v &\overset{*}{\sim} (x_1x_2\dots x_{g'''-1})a_{j+2}a_{j+1}a_j a_{j-1}\\&\quad (x_{g'''+1}\dots x_{g''-1})(x_{g''+1}\dots x_{g'-1})(x_{g'+1}\dots x_{g-1}).\end{aligned}$$

Continuing to the above process, we can obtain the maximum integer h with $j+1 \le h \le k$, a left factor v_1' of v as well as a words $v_2' \in A^*$ such that

$$v \overset{*}{\sim} v_1'\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2'.$$

In the first case that the letters $a_{j-1}, a_j, \dots, a_h$ do not appear in v_1', we observe that a_{h+1} does not appear in v_1' also. Otherwise, suppose that $v_1' = x_1x_2\dots x_l$ in which $x_{l'}$ is the last appearance of the letter a_{h+1}. Then it is certain that $a_{j-1}, a_j, \dots, a_{h+1}$ do not appear in $x_{l'+1}x_{l'+2}\dots x_l$, and hence $a_{j-1}, a_j, a_{j+1}, \dots, a_h$ commutes with the letters appearing in $x_{l'+1}x_{l'+2}\dots x_l$. Contradicting to the definition of the integer h, this implies that

$$\begin{aligned}v &\overset{*}{\sim} v_1'\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2'\\&\overset{*}{\sim} x_1x_2\dots x_{l'}\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\\&\quad x_{l'+1}x_{l'+2}\dots x_l v_2'\\&= x_1x_2\dots x_{l'-1}\,(a_{h+1}a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,x_{l'+1}\\&\quad x_{l'+2}\dots x_l v_2'.\end{aligned}$$

Following by the observation that a_{h+1} does not appear in v_1', we can see that the letters $a_{j-1}, a_j, a_{j+1}, \dots, a_h$ commutes with those in v_i'. Putting $v_1 = \epsilon$ and $v_2 = v_1'v_2'$, we have

$$\begin{aligned}v &\overset{*}{\sim} v_1'\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2'\\&\overset{*}{\sim} (a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_1'v_2'\\&= v_1\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2.\end{aligned}$$

In the second case that some of the letters $a_{j-1}, a_j, \dots, a_h$ appears in v_1', suppose that $v_1' = x_1x_2\dots x_l$ in which $x_{l'}$ is the last one of the appearances of $a_{j-1}, a_j, \dots, a_h$. Then the letter a_{h+1} does not appear in $x_{l'+1}x_{l'+2}\dots x_l$. Otherwise, under the assumption that $x_{l''}$ is the last appearance of a_{h+1} in $x_{l'+1}x_{l'+2}\dots x_l$, the letters $a_{j-1}, a_j, \dots, a_h$ commutes with those in $x_{l''+1}x_{l''+2}\dots x_l$. Contradicting to the definition of the integer h, this implies that

$$\begin{aligned}v &\overset{*}{\sim} v_1'\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2'\\&\overset{*}{\sim} x_1x_2\dots x_{l''}\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\\&\quad x_{l''+1}x_{l''+2}\dots x_l v_2'\\&= x_1x_2\dots x_{l''-1}\,(a_{h+1}a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\\&\quad x_{l''+1}x_{l''+2}\dots x_l v_2'.\end{aligned}$$

Hence a_{h+1} does not appear in $x_{l'+1}x_{l'+2}\dots x_l$, so that the letters $a_{j-1}, a_j, \dots, a_h$ commutes with those in $x_{l'+1}x_{l'+2}\dots x_l$. Put $v_1 = x_1x_2\dots x_{l'}$ and $v_2 = x_{l'+1}x_{l'+2}\dots x_l v_2'$. Observing from Lemma 5.6 that a_{j-1} does not appear in $x_1x_2\dots x_{g-1}$, we claim that v_1 is a left factor of v terminated by the letter $x_{l'}$ in $a_j, a_{j+1}, \dots, a_h$ and

$$\begin{aligned}v &\overset{*}{\sim} v_1'\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2'\\&\overset{*}{\sim} x_1x_2\dots x_{l'}\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\\&\quad x_{l'+1}x_{l'+2}\dots x_l v_2'\\&= v_1\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2.\end{aligned}$$

This finishes the proof. ■

Lemma 5.9: The letter a_j does not appear in the word $x_1x_2\dots x_{g-1}$.

Proof: Otherwise, it follows from Lemma 5 that there exist an integer h with $j+1 \le h \le k$ such that

$$v \overset{*}{\sim} v_1\,(a_h a_{h-1}\dots a_{j+1}a_j a_{j-1})\,v_2$$

for some words $v_1, v_2 \in A^*$ in which v_1 is either the empty word or a left factor of v terminated by some letter in $a_j, a_{j+1}, \ldots, a_h$. The desired claim is a consequence of the following discussions:

Case 1: $v_1 = \epsilon$. In this case, since the letter a_j commutes with $a_h, a_{h-1}, \ldots, a_{j+2}$, we have

$$\begin{aligned} w_1 v &= (u_1 u_2 \ldots u_i)\,(a_{i+1} a_i \ldots a_{j+1} a_j)\, v \\ &\overset{*}{\sim} (u_1 u_2 \ldots u_i)\,(a_{i+1} a_i \ldots a_{j+1} a_j) \\ &\quad (a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1})\, v_2 \\ &\overset{*}{\sim} (u_1 u_2 \ldots u_i)\,(a_{i+1} a_i \ldots a_{j+1}) \\ &\quad (a_{j+h} a_{j+h-1} \ldots a_{j+2})\, a_j a_{j+1} a_j a_{j-1} v_2, \end{aligned}$$

and then the swapped word

$$(u_1 u_2 \ldots u_i)\,(a_{i+1} a_i \ldots a_{j+1})\,(a_{j+h} a_{j+h-1} \ldots a_{j+2})\, a_j a_{j+1} a_j a_{j-1} v_2$$

of the factor $w_1 v$ of w'' has a factor $a_j a_{j+1} a_j$. This contradicts to Lemma 5.3(4).

Case 2: v_1 is terminated by a_h. In this case, supposing that $v_1 = v_1' a_h$, we have

$$\begin{aligned} v &\overset{*}{\sim} v_1\,(a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1})\, v_2 \\ &= v_1' a_h a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1} v_2, \end{aligned}$$

and then the swapped word $v_1' a_h a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1} v_2$ of the factor v of w'' has a factor $a_h a_h$. This contradicts to Lemma 5.3(3).

Case 3: v_1 is terminated by some letter $a_{j'}$ in $a_j, a_{j+1}, \ldots, a_{h-1}$. In this case, supposing that $v_1 = v_1' a_{j'}$, we have

$$\begin{aligned} v &\overset{*}{\sim} v_1\,(a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1})\, v_2 \\ &= v_1' a_{j'}\,(a_h a_{h-1} \ldots a_{j+1} a_j a_{j-1})\, v_2, \\ &\overset{*}{\sim} v_1'\,(a_h a_{h-1} \ldots a_{j'+2} a_{j'} a_{j'+1} a_{j'} a_{j'-1} \ldots a_{j+1} a_j a_{j-1})\, v_2 \end{aligned}$$

and then the swapped word $v_1'(a_h a_{h-1} \ldots a_{j'+2} a_{j'} a_{j'+1} a_{j'} a_{j'-1} \ldots a_{j+1} a_j a_{j-1}) v_2$ of the factor v of w'' has a factor $a_h a_h$. This also contradicts to Lemma 5.3(4).

As a consequence of the above lemma, we can see by Lemma 5.1 that the letter $x_g = a_{j-1}$ commutes with the letters $x_1, x_2, \ldots, x_{g-1}$, and hence

$$\begin{aligned} w \overset{*}{\sim} w'' &= w_1 w_2 \\ &= w_1 (x_1 x_2 \ldots x_{g-1}) a_{j-1} (x_{g+1} \ldots x_m) \\ &\overset{*}{\sim} w_1 a_{j-1} (x_1 x_2 \ldots x_{g-1})(x_{g+1} \ldots x_m). \end{aligned}$$

This completes the proof of Theorem 5.4. ■

Corollary 5.10: For any integer r with $n/2 \leq r \leq n$, the shortest r-rank input words of the automaton $\mathscr{M}_n$ have length $M(n, r)$.

6 Conclusions and discussion

The study of synchronising automata was helpful with restoring control over the devices that can be described as a synchronising automaton. Černý conjecture is the primary issue of the study of synchronising automata. The investigation on the r-rank input words of automata was motivated by the consideration on the problem to restore control over the devices that can not be described as a synchronising automaton. Pin conjecture and its restriction rank conjecture are generalisations of Černý conjecture. Of course, the research on Pin conjecture and rank conjecture are advantageous for resolving the problem to restore control over devices.

The present paper is concentrated on the r-rank input words of matching automata, given an n-state matching automaton $\mathscr{A}$ over m input letters. It is shown that $\mathscr{A}$ has an r-rank input word if and only if $n/2 \leq r \leq n$. Moreover, if $\mathscr{A}$ has an r-rank input word, then an r-rank input word of $\mathscr{A}$ having length at most $M(n, r)$ can be found in $O(nm + n^2)$ times. Since $M(n, r) \leq (n - r)^2$, this implies the conclusion that Pin conjecture and rank conjecture are true for matching automata. Furthermore, it is proved that the shortest r-rank input words of the n-state extreme matching automaton $\mathscr{M}_n$ have length $M(n, r)$, and thus the tight bound for the lengths of the shortest r-rank input words of n-state matching automata is $M(n, r)$.

References

Almeida, J., Magolis, S., Steinberg, B. and Volkov, M.V. (2009) 'Representation theory of finite semigroups, semigroup radicals and formal language theory', *Trans. Amer. Math. Sci.*, Vol. 361, No. 3, pp.1429–1461.

Almeida, J. and Steinberg, B. (2009) 'Matrix mortality and the Černý-Pin conjecture', *Lect. Notes Comput. Sci.*, Vol. 5583, pp.67–80.

Ananichev, D.S. and Volkov, M.V. (2004) 'Synchronizing monotonic automata', *Theoret. Comput. Sci.*, Vol. 327, No. 3, pp.225–239.

Ananichev, D.S. and Volkov, M.V. (2005) 'Synchronizing generalized monotonic automata', *Theoret. Comput. Sci.*, Vol. 330, Nos. 1, pp.3–13.

Ashby, W.R. (1956) *An Introduction to Cybernetics*, Chamman & Hall, London.

Benenson, Y. (2003) 'DNA molecule provides a computing machine with both data and fuel', *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 100, No. 5, pp.2191–2196.

Benenson, Y., Paz-Elizur, T. and Adar, R. (2001) 'Programmable and autonomous computing machine made of biomolecules', *Nature*, Vol. 414, No. 6862, pp.430–434.

Burgin, M., Mikkilineni, R. and Morana, G. (2016) 'Intelligent organisation of semantic networks, DIME network architecture and grid automata', *International Journal of Embedded Systems*, Vol. 8, No. 4, pp.352–366.

Černý, J. (1964) 'Poznámka k homogénnym experimentom s konečnými automatmi', *Math. Slovaca*, Vol. 14, No. 3, pp.208–16.

Cui, Z.H., He, Y. and Sun, S.Y. (2019) 'Synchronizing bounded partially ordered automata', *Chinese Journal of Computers*, Vol. 42, No. 3, pp.610–623.

Cui, Z.H., Wang, Z.X. and He, Y. (2023) 'On the synchronizing problem of tree-like partially ordered automata', *Chinese Journal of Computers*, Vol. 46, No. 9, pp.1961–1976.

de Bondt, M., Don, H. and Zantema, H. (2020) 'Slowly synchronizing automata with fixed alphabet size', *Information & Computation*, Vol. 279, p.104614.

Dubuc, L. (1998) 'Sur les automates circulaires et la conjecture de Černý', *RAIRO Inform. Theoret. Appl.*, Vol. 32, Nos. 1–3, pp.21–24.

Eppstein, D. (1990) 'Reset sequences for monotonic automata', *SIAM J. Comput.*, Vol. 19, Nos. 1–3, pp.500–510.

He, Y., Chen, X.P. and Wang, Z.X. (2021) 'Extremal synchronizing circular automata'. *Information & Computation*, Vol. 281, p.104817.

Hennie, F.C. (1964) 'Fault detecting experiments for sequence circuits', *Pro. 5th Annual Symp. Switching Circuit Theory and Logical Design*, IEEE, pp.95–110.

Hrishnaswamy, S., Plaza, S.M., Aarkov, I.L. et al. (2008) 'Signature-based SER analysis and design of logic circuits', *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 28, No. 1, pp.74–86.

Imreh, B. and Steinby, M. (1995) 'Some remarks on directable automata', *Acta Cybernetica*, Vol. 12, No. 1, pp.23–35.

Kari, J. (2001) 'A counter example to a conjecture concerning synchronizing word in finite automata', *EATCS Bulletin*, Vol. 73, pp.146–147.

Kari, J. (2003) 'Synchronising finite automata on Eulerian digraphs', *Theor. Comput. Sci.*, Vol. 259, pp.223–232.

Jürgensen H. (2008) 'Synchronization', *Information & Computation*, Vol. 206, Nos. 9–10, pp.1033–1044.

Pin, J-E. (1978a) 'Sur un cas particulier de la conjecture de Cerny', *Lect. Notes Comput. Sci.*, Vol. 62, pp.345–352.

Pin, J-E. (1978b) 'Sur les mots synchronisants dans un automate fini', *Elektronische Informationverarbeitung und Kybernetik*, Vol. 14, pp.283–289.

Pin, J-E. (1983) 'On two combinatorial problems arising from automata theory', *Ann. Discrete Math.*, Vol. 17, pp.535–548.

Rystsov, I.K. (1992) 'Rank of a finite automaton', *Cybernetics and Systems Analysis*, Vol. 28, No. 3, pp.323–328.

Salomaa, A. (2012) 'Composition sequences and synchronizing automata', *Lect. Notes Comput. Sci.*, Vol. 7160, pp.403–416.

Steinberg, B. (2011) 'The Černý conjecture for one-cluster automata with prime length cycle', *Theoret. Comput. Sci.*, Vol. 412, No. 9, pp.5487–5491.

Trahtman, A.N. (2007) 'The Černý conjecture for aperiodic automata', *Discrete Math. Theor. Comput. Sci.*, Vol. 9, No. 2, pp.3–10.

Volkov, M.V. (2009) 'Synchronizing automata preserving a chain of partial orders', *Theoret. Comput. Sci.*, Vol. 410, Nos. 1–3, pp.3513–3519.

Volkov, M.V. (2008) 'Synchronizing automata and the Černý conjecture', *Lect. Notes Comput. Sci.*, Vol. 5196, pp.11–27.