数理解析研究所講究録
第 1849 巻 2013 年 57-63

Reversible multi-head finite automata and space-bounded Turing machines

Kenichi Morita
Graduate School of Engineering, Hiroshima University
morita.rcomp@gmail.com

1 Introduction

A multi-head finite automaton is a classical model for language recognition, and has relatively high recognition capability (see [1] for the survey). In [6], a reversible two-way multi-head finite automaton is introduced, and it is conjectured that a deterministic two-way multi-head finite automaton can be simulated by a reversible one with the same number of heads. Here, we show it by giving a concrete conversion method. The technique employed here is based on the method of Lange et al. [2] where a computation tree of a deterministic space-bounded Turing machine is traversed by a reversible one using the same amount of space. But, our method is much simpler and does not assume a simulated automaton always halts, and hence the converted reversible automaton traverses a computation graph that may not be a tree. This method can be applied to a general class of deterministic Turing machines. We also show that reversible MFAs can be easily implemented by a rotary element, a kind of a reversible logic element.

2 A two-way multi-head finite automaton

Definition 1 *A* two-way multi-head finite automaton *(MFA) consists of a finite-state control, a finite number of heads that can move in two directions, and a read-only input tape (Fig. 1). An MFA with k heads is denoted by MFA(k). It is formally defined by*

$$M = (Q, \Sigma, k, \delta, \rhd, \lhd, q_0, A, R),$$

where Q is a nonempty finite set of states, Σ is a nonempty finite set of input symbols, $k\,(\in \{1, 2, \dots\})$ is a number of heads, $\rhd$ and $\lhd$ are left and right endmarkers, respectively, which are not elements of Σ (i.e., $\{\rhd, \lhd\} \cap \Sigma = \emptyset$), $q_0\,(\in Q)$ is the initial state, $A\,(\subset Q)$ is

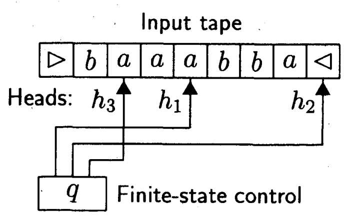

Figure 1: A two-way multi-head finite automaton (MFA).

a set of accepting states, and $R\,(\subset Q)$ *is a set of rejecting states such that* $A \cap R = \emptyset$. δ *is a subset of* $(Q \times ((\Sigma \cup \{\triangleright, \triangleleft\})^k \cup \{-1, 0, +1\}^k) \times Q)$ *that determines the transition relation on M's configurations (defined below). Note that* $-1, 0$, *and* $+1$ *stand for left-shift, no-shift, and right-shift of each head, respectively. In what follows, we also use* $-$ *and* $+$ *instead of* -1 *and* $+1$ *for simplicity. Each element* $r = [p, \mathbf{x}, q] \in \delta$ *is called a* rule *(in the triple form) of* M*, where* $\mathbf{x} = [s_1, \ldots, s_k] \in (\Sigma \cup \{\triangleright, \triangleleft\})^k$ *or* $\mathbf{x} = [d_1, \ldots, d_k] \in \{-1, 0, +1\}^k$. *A rule of the form* $[p, [s_1, \ldots, s_k], q]$ *is called a* read-rule, *and means if* M *is in the state* p *and reads symbols* $[s_1, \ldots, s_k]$ *by its* k *heads, then enter the state* q. *A rule of the form* $[p, [d_1, \ldots, d_k], q]$ *is called a* shift-rule, *and means if* M *is in the state* p *then shift the heads to the directions* $[d_1, \ldots, d_k]$ *and enter the state* q.

Suppose a word of the form $\triangleright w \triangleleft \in (\{\triangleright\}\Sigma^*\{\triangleleft\})$ is given to M. For any $q \in Q$ and for any $\mathbf{h} \in \{0, \ldots, |w|+1\}^k$, a triple $[\triangleright w \triangleleft, q, \mathbf{h}]$ is called a *configuration* of M on w. We now define a function $s_w : \{0, \ldots, |w|+1\}^k \to (\Sigma \cup \{\triangleright, \triangleleft\})^k$ as follows. If $\triangleright w \triangleleft = a_0 a_1 \cdots a_n a_{n+1}$ (hence $a_0 = \triangleright, a_{n+1} = \triangleleft$, and $w = a_1 \cdots a_n \in \Sigma^*$), and $\mathbf{h} = [h_1, \ldots, h_k] \in \{0, \ldots, |w|+1\}^k$, then $s_w(\mathbf{h}) = [a_{h_1}, \ldots, a_{h_k}]$. The function s_w gives a k-tuple of symbols in $\triangleright w \triangleleft$ read by the k heads of M at the position $\mathbf{h}$. The *transition relation* $\vdash_{M}$ between a pair of configurations is defined as follows.

$$[\triangleright w \triangleleft, q, \mathbf{h}] \vdash_{M} [\triangleright w \triangleleft, q', \mathbf{h}']$$
$$\text{iff } ([q, s_w(\mathbf{h}), q'] \in \delta \ \wedge\ \mathbf{h}' = \mathbf{h}) \ \vee\ \exists\, \mathbf{d} \in \{-1, 0, +1\}^k \ ([q, \mathbf{d}, q'] \in \delta \ \wedge\ \mathbf{h}' = \mathbf{h} + \mathbf{d})$$

The reflexive and transitive closure of the relation $\vdash_{M}$ is denoted by $\vdash_{M}^{*}$.

Definition 2 *Let* $M = (Q, \Sigma, k, \delta, \triangleright, \triangleleft, q_0, A, R)$ *be an MFA.* M *is called* deterministic *iff the following condition holds.*

$$\forall\, r_1 = [p, \mathbf{x}, q] \in \delta,\ \forall\, r_2 = [p', \mathbf{x}', q'] \in \delta$$
$$((r_1 \neq r_2 \ \wedge\ p = p') \Rightarrow (\mathbf{x} \notin \{-1, 0, +1\}^k \wedge \mathbf{x}' \notin \{-1, 0, +1\}^k \wedge \mathbf{x} \neq \mathbf{x}'))$$

It means that for any two distinct rules r_1 *and* r_2 *in* δ, *if* $p = p'$ *then they are both read-rules and the* k*-tuples of symbols* $\mathbf{x}$ *and* $\mathbf{x}'$ *are different.*

M *is called* reversible *iff the following condition holds.*

$$\forall\, r_1 = [p, \mathbf{x}, q] \in \delta,\ \forall\, r_2 = [p', \mathbf{x}', q'] \in \delta$$
$$((r_1 \neq r_2 \ \wedge\ q = q') \Rightarrow (\mathbf{x} \notin \{-1, 0, +1\}^k \wedge \mathbf{x}' \notin \{-1, 0, +1\}^k \wedge \mathbf{x} \neq \mathbf{x}'))$$

It means that for any two distinct rules r_1 *and* r_2 *in* δ, *if* $q = q'$ *then they are both read-rules and the* k*-tuples of symbols* $\mathbf{x}$ *and* $\mathbf{x}'$ *are different.*

We denote a deterministic MFA (or MFA(k)) by DMFA (or DMFA(k)), and a reversible and deterministic MFA (or MFA(k)) by RDMFA (or RDMFA(k)).

Definition 3 *Let* $M = (Q, \Sigma, k, \delta, \triangleright, \triangleleft, q_0, A, R)$ *be an MFA. We say an input word* $w \in \Sigma^*$ *is* accepted *by* M, *if* $[\triangleright w \triangleleft, q_0, \mathbf{0}] \vdash_{M}^{*} [\triangleright w \triangleleft, q, \mathbf{h}]$ *for some* $q \in A$ *and* $\mathbf{h} \in \{0, \ldots, |w|+1\}^k$, *where* $\mathbf{0} = [0, \ldots, 0] \in \{0\}^k$. *The configurations* $[\triangleright w \triangleleft, q_0, \mathbf{0}]$ *and* $[\triangleright w \triangleleft, q, \mathbf{h}]$ *such that* $q \in A$ *are called an* initial configuration *and an* accepting configuration, *respectively. The* language *accepted by* M *is the set of all words accepted by* M, *and is denoted by* $L(M)$, *i.e.,*

$$L(M) = \{w \mid \exists q \in A, \exists \mathbf{h} \in \{0, \ldots, |w|+1\}^k([\triangleright w \triangleleft, q_0, \mathbf{0}] \vdash_{M}^{*} [\triangleright w \triangleleft, q, \mathbf{h}])\}.$$

Lemma 1 [6] *Let* $M = (Q, \Sigma, k, \delta, \triangleright, \triangleleft, q_0, A, R)$ *be an RDMFA. If the initial state of* M *does not appear as the third component of any rule, then* M *eventually halts for any input* $w \in \Sigma^*$.

3 Converting a DMFA(k) into an RDMFA(k)

We show that for any given DMFA(k) M we can construct an RDMFA(k) $M^\dagger$ that simulates M. Here, we make $M^\dagger$ so that it traverses a computation graph from the node that corresponds to the initial configuration. Note that, if M halts on an input w, then the computation graph becomes a finite tree. But, if it loops, then the graph is not a tree. We shall see that both cases are managed properly.

Theorem 1 *For any DMFA(k)* $M = (Q, \Sigma, k, \delta, \rhd, \lhd, q_0, A, R)$, *we can construct an RDMFA(k)* $M^\dagger = (Q^\dagger, \Sigma, k, \delta^\dagger, \rhd, \lhd, q_0, \{\hat{q}_0^1\}, \{q_0^1\})$ *that satisfies the following.*

$$\forall w \in \Sigma^* \; (w \in L(M) \;\Rightarrow\; [\rhd w \lhd, q_0, \mathbf{0}] \;\overset{*}{\underset{M^\dagger}{\vdash}}\; [\rhd w \lhd, \hat{q}_0^1, \mathbf{0}])$$
$$\forall w \in \Sigma^* \; (w \notin L(M) \;\Rightarrow\; [\rhd w \lhd, q_0, \mathbf{0}] \;\overset{*}{\underset{M^\dagger}{\vdash}}\; [\rhd w \lhd, q_0^1, \mathbf{0}])$$

Proof outline. We first define the *computation graph* $G_{M,w} = (V, E)$ of M with an input $w \in \Sigma^*$ as follows. Let C be the set of all configurations of M with w, i.e., $C = \{[\rhd w \lhd, q, \mathbf{h}] \mid q \in Q \wedge \mathbf{h} \in \{0, \ldots, |w|+1\}^k\}$. The set $V (\subset C)$ of nodes is the smallest set that contains the initial configuration $[\rhd w \lhd, q_0, \mathbf{0}]$, and satisfies the following condition: $\forall c_1, c_2 \in C\, ((c_1 \in V \wedge (c_1 \underset{M}{\vdash} c_2 \vee c_2 \underset{M}{\vdash} c_1)) \Rightarrow c_2 \in V)$. The set E of directed edges is: $E = \{(c_1, c_2) \mid c_1, c_2 \in V \wedge c_1 \underset{M}{\vdash} c_2\}$. Apparently $G_{M,w}$ is a finite connected graph. Since M is deterministic, outdegree of each node in V is either 0 or 1, where a node of outdegree 0 corresponds to a halting configuration. It is easy to see there is at most one node of outdegree 0 in $G_{M,w}$, and if there is one, then $G_{M,w}$ is a tree (Fig. 2 (a)). On the other hand, if there is no node of outdegree 0, then the graph represents the computation of M having a loop, and thus it is not a tree (Fig. 2 (b)).

Here we assume M performs read and shift operations alternately. Hence, Q is written as $Q = Q_{\text{read}} \cup Q_{\text{shift}}$ for some Q_{read} and Q_{shift} such that $Q_{\text{read}} \cap Q_{\text{shift}} = \emptyset$, and δ satisfies the following condition:

$$\forall\, [p, \mathbf{x}, q] \in \delta \; (\mathbf{x} \in (\Sigma \cup \{\rhd, \lhd\})^k \Rightarrow p \in Q_{\text{read}} \wedge q \in Q_{\text{shift}}) \;\wedge$$
$$\forall\, [p, \mathbf{x}, q] \in \delta \; (\mathbf{x} \in \{-, 0, +\}^k \Rightarrow p \in Q_{\text{shift}} \wedge q \in Q_{\text{read}}).$$

We define the following five functions.

$$\begin{aligned}
\text{prev-read}(q) &= \{[p, \mathbf{d}] \mid p \in Q_{\text{shift}} \wedge \mathbf{d} \in \{-, 0, +\}^k \wedge [p, \mathbf{d}, q] \in \delta\} \\
\text{prev-shift}(q, \mathbf{s}) &= \{p \mid p \in Q_{\text{read}} \wedge [p, \mathbf{s}, q] \in \delta\} \\
\text{deg}_{\text{r}}(q) &= |\text{prev-read}(q)| \\
\text{deg}_{\text{s}}(q, \mathbf{s}) &= |\text{prev-shift}(q, \mathbf{s})| \\
\text{deg}_{\text{max}}(q) &= \begin{cases} \text{deg}_{\text{r}}(q) & \text{if } q \in Q_{\text{read}} \\ \max\{\text{deg}_{\text{s}}(q, \mathbf{s}) \mid \mathbf{s} \in (\Sigma \cup \{\rhd, \lhd\})^k\} & \text{if } q \in Q_{\text{shift}} \end{cases}
\end{aligned}$$

Assume M is in the configuration $[\rhd w \lhd, q, \mathbf{h}]$. If q is a read-state (shift-state, respectively), then $\text{deg}_{\text{r}}(q)$ ($\text{deg}_{\text{s}}(q, s_w(\mathbf{h}))$) denotes the total number of previous configurations of $[\rhd w \lhd, q, \mathbf{h}]$, and each element $[p, \mathbf{d}] \in \text{prev-read}(q)$ ($p \in \text{prev-shift}(q, s_w(\mathbf{h}))$) gives a previous state and a shift direction (a previous state). We further assume that the set Q and, of course, the set $\{-1, 0, +1\}$ are totally ordered, and thus the elements of the sets $\text{prev-read}(q)$ and $\text{prev-shift}(q, s)$ are sorted based on these orders. So, in the following, we denote $\text{prev-read}(q)$ and $\text{prev-shift}(q, s)$ by the ordered lists as below.

$$\begin{aligned}
\text{prev-read}(q) &= [[p_1, \mathbf{d}_1], \ldots, [p_{\text{deg}_{\text{r}}(q)}, \mathbf{d}_{\text{deg}_{\text{r}}(q)}]] \\
\text{prev-shift}(q, \mathbf{s}) &= [p_1, \ldots, p_{\text{deg}_{\text{s}}(q, \mathbf{s})}]
\end{aligned}$$

We now construct an RDMFA(k) $M^\dagger$ that simulates the DMFA(k) M by traversing $G_{M,w}$ for a given w. First, $Q^\dagger$ is as below.

$$Q^\dagger = \{q, \hat{q} \mid q \in Q\} \cup \{q^j, \hat{q}^j \mid q \in Q \wedge j \in (\{1\} \cup \{1, \ldots, \mathrm{deg_{max}}(q)\})\}$$

$\delta^\dagger$ is given as below, where $\mathbf{S} = (\Sigma \cup \{\rhd, \lhd\})^k$.

$$\begin{aligned}
\delta^\dagger =\ & \delta_1 \cup \cdots \cup \delta_6 \cup \hat{\delta}_1 \cup \cdots \cup \hat{\delta}_5 \cup \delta_{\mathrm{a}} \cup \delta_{\mathrm{r}} \\
\delta_1 =\ & \{ [p_1, \mathbf{d}_1, q^2], \ldots, [p_{\mathrm{deg_r}(q)-1}, \mathbf{d}_{\mathrm{deg_r}(q)-1}, q^{\mathrm{deg_r}(q)}], [p_{\mathrm{deg_r}(q)}, \mathbf{d}_{\mathrm{deg_r}(q)}, q] \mid \\
& \quad q \in Q_{\mathrm{read}} \wedge \mathrm{deg_r}(q) \geq 1 \wedge \text{prev-read}(q) = [[p_1, \mathbf{d}_1], \ldots, [p_{\mathrm{deg_r}(q)}, \mathbf{d}_{\mathrm{deg_r}(q)}]] \} \\
\delta_2 =\ & \{ [p_1, \mathbf{s}, q^2], \ldots, [p_{\mathrm{deg_s}(q,\mathbf{s})-1}, \mathbf{s}, q^{\mathrm{deg_s}(q,\mathbf{s})}], [p_{\mathrm{deg_s}(q,\mathbf{s})}, \mathbf{s}, q] \mid \\
& \quad q \in Q_{\mathrm{shift}} \wedge \mathbf{s} \in \mathbf{S} \wedge \mathrm{deg_s}(q, \mathbf{s}) \geq 1 \wedge \text{prev-shift}(q, \mathbf{s}) = [p_1, \ldots, p_{\mathrm{deg_s}(q,\mathbf{s})}] \} \\
\delta_3 =\ & \{ [q^1, -\mathbf{d}_1, p_1^1], \ldots, [q^{\mathrm{deg_r}(q)}, -\mathbf{d}_{\mathrm{deg_r}(q)}, p^1_{\mathrm{deg_r}(q)}] \mid \\
& \quad q \in Q_{\mathrm{read}} \wedge \mathrm{deg_r}(q) \geq 1 \wedge \text{prev-read}(q) = [[p_1, \mathbf{d}_1], \ldots, [p_{\mathrm{deg_r}(q)}, \mathbf{d}_{\mathrm{deg_r}(q)}]] \} \\
\delta_4 =\ & \{ [q^1, \mathbf{s}, p_1^1], \ldots, [q^{\mathrm{deg_s}(q,\mathbf{s})}, \mathbf{s}, p^1_{\mathrm{deg_s}(q,\mathbf{s})}] \mid \\
& \quad q \in Q_{\mathrm{shift}} \wedge \mathbf{s} \in \mathbf{S} \wedge \mathrm{deg_s}(q, \mathbf{s}) \geq 1 \wedge \text{prev-shift}(q, \mathbf{s}) = [p_1, \ldots, p_{\mathrm{deg_s}(q,\mathbf{s})}] \} \\
\delta_5 =\ & \{ [q^1, \mathbf{s}, q] \mid q \in Q_{\mathrm{shift}} - (A \cup R) \wedge \mathbf{s} \in \mathbf{S} \wedge \mathrm{deg_s}(q, \mathbf{s}) = 0 \} \\
\hat{\delta}_i =\ & \{ [\hat{p}, \mathbf{x}, \hat{q}] \mid [p, \mathbf{x}, q] \in \delta_i \} \ (i = 1, \ldots, 5) \\
\delta_6 =\ & \{ [q, \mathbf{s}, q^1] \mid q \in Q_{\mathrm{read}} - \{q_0\} \wedge \mathbf{s} \in \mathbf{S} \wedge \neg \exists p\, ([q, \mathbf{s}, p] \in \delta) \} \\
\delta_{\mathrm{a}} =\ & \{ [q, \mathbf{0}, \hat{q}^1] \mid q \in A \} \\
\delta_{\mathrm{r}} =\ & \{ [q, \mathbf{0}, q^1] \mid q \in R \}
\end{aligned}$$

The set of states $Q^\dagger$ has four types of states. They are of the forms $q, \hat{q}, q^j$ and $\hat{q}^j$. The states without a superscript (i.e., q and $\hat{q}$) are for forward computation, while those with a superscript (i.e., q^j and $\hat{q}^j$) are for backward computation. Note that $Q^\dagger$ contains q^1 and $\hat{q}^1$ even if $\mathrm{deg_{max}}(q) = 0$. The states with "^" (i.e., $\hat{q}$ and $\hat{q}^j$) are the ones indicating that an accepting configuration was found in the process of traverse, while those without "^" (i.e., q and q^j) are for indicating no accepting configuration has been found so far.

The set of rules δ_1 (δ_2, respectively) is for simulating forward computation of M in $G_{M,w}$ for M's shift-states (read-states). δ_3 (δ_4, respectively) is for simulating backward computation of M in $G_{M,w}$ for M's read-states (shift-states). δ_5 is for turning the direction of computation from backward to forward in $G_{M,w}$ for shift-states. $\hat{\delta}_i$ ($i = 1, \ldots, 5$) is the set of rules for the states of the form $\hat{q}$, and is identical to δ_i except that each state has "^". δ_6 is for turning the direction of computation from forward to backward in $G_{M,w}$ for halting configurations with a read-state. δ_{a} (δ_{r}, respectively) is for turning the direction of computation from forward to backward for accepting (rejecting) states. Each rule in δ_{a} makes $M^\dagger$ change the state from a one without "^" to the corresponding one with "^". We can verify that $M^\dagger$ is deterministic and reversible by a careful inspection of $\delta^\dagger$.

$M^\dagger$ simulates M as follows. First, consider the case $G_{M,w}$ is a tree. If an input w is given, $M^\dagger$ traverses $G_{M,w}$ by the depth-first search (Fig. 2 (a)). Note that the search starts not from the root of the tree but from the leaf node $[\rhd w \lhd, q_0, \mathbf{0}]$. Since each node of $G_{M,w}$ is identified by the configuration of M of the form $[\rhd w \lhd, q, \mathbf{h}]$, it is easy for $M^\dagger$ to keep it by the configuration of $M^\dagger$. But, if $[\rhd w \lhd, q, \mathbf{h}]$ is a non-leaf node, it may be visited $\mathrm{deg_{max}}(q) + 1$ times (i.e., the number of its child nodes plus 1) in the process of depth-first search, and thus $M^\dagger$ should keep this information in the finite state control. To do so, $M^\dagger$ uses $\mathrm{deg_{max}}(q) + 1$ states $q^1, \ldots, q^{\mathrm{deg_{max}}(q)}$, and q for each state q of M. Here, the states $q^1, \ldots, q^{\mathrm{deg_{max}}(q)}$ are used for visiting its child nodes, and q is used for visiting its parent node. In other words, the states with a superscript are for going downward in the tree (i.e., a backward simulation of M), and the state without a superscript is for going upward in the tree (i.e., a forward simulation). At a leaf node $[\triangleright w \triangleleft, q, \mathbf{h}]$, which satisfies $\deg_s(q, s_w(\mathbf{h})) = 0$, $M^\dagger$ turns the direction of computing by the rule $[q^1, s_w(\mathbf{h}), q] \in \delta_5$.

If $M^\dagger$ enters an accepting state of M, say q_a, which is the root of the tree while traversing the tree, then $M^\dagger$ goes to the state $\hat{q}_a$, and continues the depth-first search. After that, $M^\dagger$ uses the states of the form $\hat{q}$ and $\hat{q}^j$ indicating that the input w should be accepted. $M^\dagger$ will eventually reach the initial configuration of M by its configuration $[\triangleright w \triangleleft, \hat{q}_0^1, \mathbf{0}]$. Thus, $M^\dagger$ halts and accepts the input. Note that we can assume there is no rule of the form $[q_0, \mathbf{s}, q]$ such that $\mathbf{s} \notin \{\triangleright\}^k$ in δ, because the initial configuration of M is $[\triangleright w \triangleleft, q_0, \mathbf{0}]$. Therefore, $M^\dagger$ never reaches a configuration $[\triangleright w \triangleleft, q_0, \mathbf{h}]$ of M such that $\mathbf{h} \neq \mathbf{0}$. If $M^\dagger$ enters a halting state of M other than the accepting states, then it continues the depth-first search without entering a state of the form $\hat{q}$. Also in this case, $M^\dagger$ will finally reach the initial configuration of M by its configuration $[\triangleright w \triangleleft, q_0^1, \mathbf{0}]$. Thus, $M^\dagger$ halts and rejects the input.

Second, consider the case $G_{M,w}$ is not a tree (Fig. 2 (b)). In this case, since there is no accepting configuration in $G_{M,w}$, $M^\dagger$ never enters an accepting state of M no matter how $M^\dagger$ visits the nodes of $G_{M,w}$. Thus, $M^\dagger$ uses only the states without "^". From $\delta^\dagger$ we can see q_0^1 is the only halting state without "^". By Lemma 1, $M^\dagger$ must halt with the configuration $[\triangleright w \triangleleft, q_0^1, \mathbf{0}]$, and rejects the input. By above, we have the theorem. □

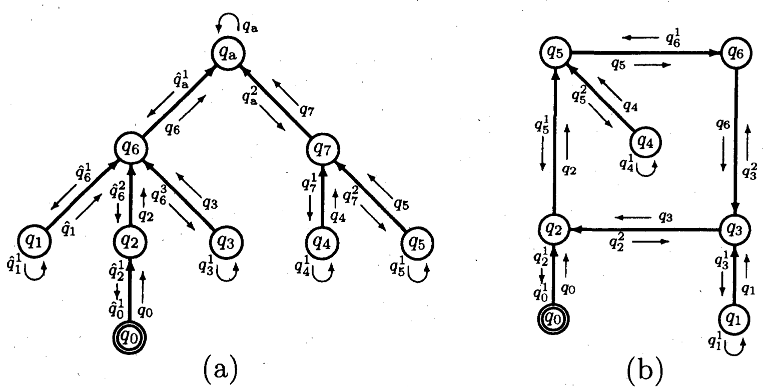

Figure 2: Examples of computation graphs $G_{M,w}$ of a DMFA(k) M. Each node represents a configuration of M, though only a state of the finite-state control is written in a circle. Thick arrows are the edges of $G_{M,w}$. The node labeled by q_0 represents the initial configuration of M. An RDMFA(k) $M^\dagger$ traverses these graphs along thin arrows using its configurations. (a) This is a case M halts in an accepting state q_a. Here, the state transition of $M^\dagger$ in the traversal of the tree is as follows: $q_0 \to q_2 \to q_6^3 \to q_3^1 \to q_3 \to q_6 \to q_a^2 \to q_7^1 \to q_4^1 \to q_4 \to q_7^2 \to q_5^1 \to q_5 \to q_7 \to q_a \to \hat{q}_a^1 \to \hat{q}_6^1 \to \hat{q}_1^1 \to \hat{q}_1 \to \hat{q}_6^2 \to \hat{q}_2^1 \to \hat{q}_0^1$. (b) This is a case M loops forever. Here, $M^\dagger$ traverses the graph as follows: $q_0 \to q_2^2 \to q_3^1 \to q_1^1 \to q_1 \to q_3^2 \to q_6^1 \to q_5^1 \to q_2^1 \to q_0^1$.

4 Applying the method to Turing machines

It has been shown by Lange et al. [2] that DSPACE($S(n)$) = RDSPACE($S(n)$) holds for any space function $S(n)$. However, by applying the method of the previous section, we can convert a deterministic Turing machine to a reversible one very easily. By this, we can obtain a slightly stronger result by a much simpler method. (Here, we omit its proof.)

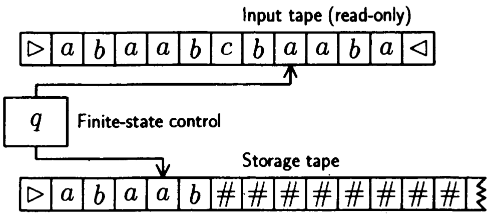

Figure 3: A two-tape Turing machine.

Definition 4 *A* two-tape Turing machine *(TM) consists of a finite-state control with two heads, a read-only input tape, and a storage tape (Fig. 3) . It is defined by*

$$T = (Q, \Sigma, \Gamma, \delta, \triangleright, \triangleleft, q_0, \#, A, R),$$

where Q is a nonempty finite set of states, Σ and Γ are nonempty finite sets of input symbols and storage tape symbols. $\triangleright$ and $\triangleleft$ are left and right endmarkers such that $\{\triangleright, \triangleleft\} \cap (\Sigma \cup \Gamma) = \emptyset$, where only $\triangleright$ is used for the storage tape. $q_0 (\in Q)$ is the initial state, $\# (\notin \Gamma)$ is a blank symbol of the storage tape, $A (\subset Q)$ and $R (\subset Q)$ are sets of accepting and rejecting states such that $A \cap R = \emptyset$. δ is a subset of $(Q \times (((\Sigma \cup \{\triangleright, \triangleleft\}) \times (\Gamma \cup \{\triangleright, \#\})^2) \cup \{-1, 0, +1\}^2) \times Q)$ that determines the transition relation on T's configurations. Each element $r = [p, x, y, q] \in \delta$ is called a rule *(in the quadruple form) of T, where $(x, y) = (s_1, [s_2, s_3]) \in ((\Sigma \cup \{\triangleright, \triangleleft\}) \times (\Gamma \cup \{\triangleright, \#\})^2)$ or $(x, y) = (d_1, d_2) \in \{-1, 0, +1\}^2$. A rule of the form $[p, s_1, [s_2, s_3], q]$ is called a* read-write-rule, *and means if T is in the state p and reads an input symbol s_1 and a storage tape symbol s_2, then rewrites s_2 to s_3 and enters the state q. A rule of the form $[p, d_1, d_2, q]$ is called a* shift-rule, *and means if T is in the state p then shift the two heads to the directions d_1 and d_2, and enter the state q. Determinism and reversibility of T are defined similarly as in the case of MFAs.*

Theorem 2 *For any DTM $T = (Q, \Sigma, \Gamma, \delta, \triangleright, \triangleleft, q_0, \#, A, R)$, we can construct an RDTM $T^\dagger = (Q^\dagger, \Sigma, \Gamma, \delta^\dagger, \triangleright, \triangleleft, q_0, \#, \{\hat{q}_0^1\}, \{q_0^1\})$ such that the following holds.*

$$\forall w \in \Sigma^* \quad (w \in L(T) \;\Rightarrow\; [\triangleright w \triangleleft, \triangleright, q_0, 0, 0] \mid\!\frac{*}{T^\dagger}\; [\triangleright w \triangleleft, \triangleright, \hat{q}_0^1, 0, 0])$$

$$\forall w \in \Sigma^* \quad (w \notin L(T) \wedge T \text{ with } w \text{ uses bounded amount of the storage tape}$$
$$\Rightarrow\; [\triangleright w \triangleleft, \triangleright, q_0, 0, 0] \mid\!\frac{*}{T^\dagger}\; [\triangleright w \triangleleft, \triangleright, q_0^1, 0, 0])$$

$$\forall w \in \Sigma^* \quad (w \notin L(T) \wedge T \text{ with } w \text{ uses unbounded amount of the storage tape}$$
$$\Rightarrow\; T^\dagger\text{'s computation starting from } [\triangleright w \triangleleft, \triangleright, q_0, 0, 0] \text{ does not halt})$$

Furthermore, if T uses at most m squares of the storage tape on an input w, then $T^\dagger$ with w also uses at most m squares in any of its configuration in its computing process.

5 Reversible logic circuits that simulate RDMFAs

It is possible to implement an RDMFA using only rotary elements as in the case of a reversible Turing machine [3, 4, 5]. A rotary element [3] is a reversible logic element with 4 input and 4 output lines, and 2 states shown in Fig. 4. In [3, 5], a construction method of a finite control unit and a tape square unit of a reversible Turing machine out of rotary elements is given. Though a similar method can also be applied for constructing an RMFA, accessing a tape square by many heads should be managed properly. Here, we show an example of the circuit without giving a detailed explanation.

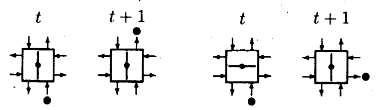

Figure 4: Operation of a rotary element. The case where the directions of the bar and the comimg signal are parallel (left), and the case where they are orthogonal (right).

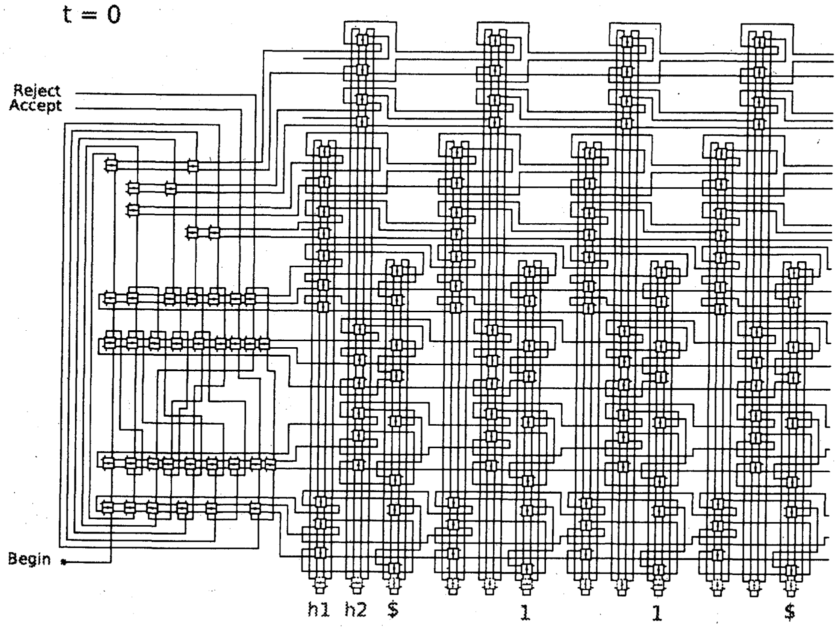

Figure 5: A circuit composed only of rotary elements that simulates the RMFA(2) M'_{2^m}.

Consider the RDMFA(2) M_{2^m} in the quadruple form that accepts $L_{2^m} = \{1^n \mid n = 2^m \text{ for some } m \in \{0, 1, \ldots\}\}$, where \$ is used as both left and right end-markers.

$M_{2^m} = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_a, q_r\}, \{1\}, 2, \delta_{2^m}, \$, \$, q_0, \{q_a\}, \{q_r\})$

$\delta_{2^m} = \{$ (1) $[q_0, [\$, \$], [0, +], q_1]$, (2) $[q_1, [\$, 1], [0, +], q_1]$, (3) $[q_1, [\$, \$], [+, -], q_2]$,
(4) $[q_2, [1, 1], [0, -], q_3]$, (5) $[q_2, [1, \$], [-, +], q_4]$, (6) $[q_2, [\$, \$], [0, 0], q_r]$,
(7) $[q_3, [1, 1], [+, -], q_2]$, (8) $[q_3, [1, \$], [-, 0], q_5]$, (9) $[q_4, [1, 1], [-, +], q_4]$,
(10) $[q_4, [\$, 1], [+, -], q_2]$, (11) $[q_5, [\$, \$], [0, 0], q_a]$, (12) $[q_5, [1, \$], [0, 0], q_r]$ $\}$

Fig. 5 shows the whole circuit of M_{2^m} for the input 1^2. Giving a particle at the Begin port in Fig. 5, the circuit starts to simulate M_{2^m}. The particle finally comes out from the output port Accept since $1^2 \in L_{2^m}$. If an input $1^n \notin L_{2^m}$ is given, the particle will appear from the Reject port.

References

[1] Holzer, M., Kutrib, M., Malcher, A.: Complexity of multi-head finite automata: Origins and directions. Theoret. Comput. Sci. 412, 83–96 (2011)

[2] Lange, K.J., McKenzie, P., Tapp, A.: Reversible space equals deterministic space. J. Comput. Syst. Sci. 60, 354–367 (2000)

[3] Morita, K.: A simple reversible logic element and cellular automata for reversible computing. In: Proc. 3rd Int. Conf. on Machines, Computations, and Universality, *LNCS 2055*. pp. 102–113. Springer-Verlag (2001)

[4] Morita, K.: Reversible computing and cellular automata — A survey. Theoret. Comput. Sci. 395, 101–131 (2008)

[5] Morita, K.: Constructing a reversible Turing machine by a rotary element, a reversible logic element with memory. Hiroshima University Institutional Repository, http://ir.lib.hiroshima-u.ac.jp/00029224 (2010)

[6] Morita, K.: Two-way reversible multi-head finite automata. Fundamenta Informaticae 110, 241–254 (2011)