ISIT'2013 1569713801

On Antidictionary Coding Based on Compacted Substring Automaton

Takahiro Ota
Dept. of Computer & Systems Engineering
Nagano Prefectural Institute of Technology
Ueda, Nagano 386-1211, Japan
Email: ota@pit-nagano.ac.jp

Hiroyoshi Morita
Grad. School of Information Systems
University of Electro-Communications
Chofu, Tokyo 182-8585, Japan
Email: morita@is.uec.ac.jp

***Abstract*—Lossless data compression via substring enumeration (CSE) has been proposed by Dubé and Beaudoin in 2010. The CSE outputs its encoder called compacted substring automaton as a codeword, and an efficient representation of the automaton is also proposed. In this paper, we prove an isomorphism between compacted substring automaton and antidictionary automaton, which is an encoder of antidictionary coding. Then we propose a new static antidictionary coding which uses the representation of compacted substring automaton instead of antidictionary. Moreover, we prove an asymptotic optimality of the proposed coding for a stationary ergodic source.**

I. Introduction

An antidictionary coding is a lossless data compression using an antidictionary automaton which accepts all strings that contain no strings of antidictionary as their substrings. An antidictionary is a set of minimal forbidden words of an input string, and an antidictionary automaton was originally introduced by Crochemore *et al.* in [1]. Various antidictionary codes have been proposed [2]–[4], and the asymptotic optimality for a proper Markov source has been proved [5]. Lossless data compression via substring enumeration (CSE) [6] is also a lossless data compression algorithm using a compacted substring automaton that accepts all substrings of circular string of an input binary string. The compression algorithm constructs a compacted automaton representation of a binary input string as its codeword. The asymptotic optimality for sequences satisfying a proper condition on stationary ergodic source has been proved [7].

In a static antidictionary coding, an antidictionary of a given input string is output as a codeword. An efficient representation of the antidictionary by means of a recursive tree representation has been proposed [2], however, the cost is too large to represent the antidictionary in practice.

In this paper, we first prove an isomorphism between both the encoders that are compacted substring automaton and antidictionary automaton. Then we propose a new antidictionary code which outputs an efficient representation of compacted substring automaton as its codeword instead of antidictionary. Moreover, we evaluate probability of a set of sequences not-satisfying the proper condition shown in [6] for the CSE algorithm on a stationary ergodic source. We prove that asymptotic optimality of the proposed algorithm for a stationary ergodic source.

II. Basic Notations and Definitions

Let $\mathcal{X}$ be a binary source alphabet $\{0,1\}$. Let $\mathcal{X}^*$ be the set of all finite strings over $\mathcal{X}$, including the empty string of length zero, denoted by λ, and let $\mathcal{X}^+$ be $\mathcal{X}^*\backslash\{\lambda\}$. For convenience, we define $\bar{a} = 1 - a$ for $a \in \mathcal{X}$.

The length of a string $\boldsymbol{x}$ is denoted as $|\boldsymbol{x}|$. We also use $|\cdot|$ to represent the cardinality of a set. For a given string $\boldsymbol{x}$, by letting $n{=}|\boldsymbol{x}|$, a substring $\boldsymbol{x}_i^j$ is defined as

$$\boldsymbol{x}_i^j = \begin{cases} x_i \cdots x_j & (1 \le i \le j \le n), \\ \lambda & (i > j). \end{cases} \tag{1}$$

Hereinafter, with no notice, we assume that the length of $\boldsymbol{x}$ is always given as n and $n \ge 2$. Therefore, the substring $\boldsymbol{x}_1^n$ of $\boldsymbol{x}$ equals $\boldsymbol{x}$. A dictionary $\mathcal{D}(\boldsymbol{x})$ is defined as the set of all the substrings of $\boldsymbol{x}$, that is,

$$\mathcal{D}(\boldsymbol{x}) = \{\boldsymbol{x}_i^j | 1 \le i \le j \le n\} \cup \{\lambda\}. \tag{2}$$

Let $\mathcal{P}(\boldsymbol{x})$ and $\mathcal{S}(\boldsymbol{x})$ respectively denote the set of all the prefixes and the suffixes of $\boldsymbol{x}$.

$$\mathcal{P}(\boldsymbol{x}) = \{\boldsymbol{x}_1^i | 1 \le i \le n\} \cup \{\lambda\}, \tag{3}$$

$$\mathcal{S}(\boldsymbol{x}) = \{\boldsymbol{x}_j^n | 1 \le j \le n\} \cup \{\lambda\}. \tag{4}$$

For a given $\boldsymbol{x}_1^n$, let $\pi(\boldsymbol{x}_1^n)$ and $\sigma(\boldsymbol{x}_1^n)$ respectively denote $\boldsymbol{x}_1^{n-1} \in \mathcal{P}(\boldsymbol{x}_1^n)$ and $\boldsymbol{x}_2^n \in \mathcal{S}(\boldsymbol{x}_1^n)$. For convenience, we define $\pi(\lambda) = \sigma(\lambda) = \lambda$. For a non-negative integer k, let $\pi^k(\boldsymbol{x}_1^n)$ and $\sigma^k(\boldsymbol{x}_1^n)$ be the functions $\pi(\cdot)$ and $\sigma(\cdot)$ applied k times to a string $\boldsymbol{x}_1^n$, where $\pi^0(\boldsymbol{x}_1^n) = \sigma^0(\boldsymbol{x}_1^n) = \boldsymbol{x}_1^n$.

A. Necklace

For a given string $\boldsymbol{x} \in \mathcal{X}^+$ and $k \ge 1$, let $\boldsymbol{x}^{\langle k \rangle}$ be a string that is generated by concatenating $\boldsymbol{x}$ followed by $\boldsymbol{x}^{\langle k-1 \rangle}$ where $\boldsymbol{x}^{\langle 0 \rangle} = \lambda$. As an example, for $k = 2$, $\boldsymbol{x}^{\langle 2 \rangle} = \boldsymbol{x}\boldsymbol{x} = x_1 \ldots x_n x_1 \ldots x_n$. A necklace of $\boldsymbol{x}$ is defined as $\boldsymbol{x}^{\langle \infty \rangle}$. In other words, the necklace of $\boldsymbol{x}$ is a circular string for $\boldsymbol{x}$.

We give Condition 1 for $\boldsymbol{x}$. Hereinafter, with no notice, we assume that $\boldsymbol{x}$ satisfies Condition 1 in the following discussions except Section V.

Condition 1. $|\{\boldsymbol{u} \in \mathcal{D}(\sigma(\boldsymbol{x}^{\langle 2 \rangle})) \,|\, n = |\boldsymbol{u}|\}| = n.$

In other words, for $1 \le i \ne j \le n$, a substring of length n starting from i in $\boldsymbol{x}^{\langle 2 \rangle}$ is not equal to a string of length n starting from j in $\boldsymbol{x}^{\langle 2 \rangle}$. Moreover, $\boldsymbol{x}$ that satisfies Condition 1

is also called *primitive* [8]. For example, $\boldsymbol{x} = 1001$ satisfies Condition 1 although $\boldsymbol{x} = 0101$ does not satisfy Condition 1.

Let $\mathcal{B}_n$ be the set of all the strings that satisfy Condition 1 in $\mathcal{X}^n$. For $\boldsymbol{x} \in \mathcal{B}_n$,

$$\mathcal{W}_n(\boldsymbol{x}) = \{\boldsymbol{u} \,|\, |\boldsymbol{u}| = n, \boldsymbol{u} \in \mathcal{D}(\sigma(\boldsymbol{x}^{\langle 2 \rangle}))\}. \tag{5}$$

The size of $\mathcal{W}_n(\boldsymbol{x})$ is equal to n since $\boldsymbol{x}$ satisfies Condition 1. The set of all the substrings of length n of $\boldsymbol{x}^{\langle \infty \rangle}$ is equal to $\mathcal{W}_n(\boldsymbol{x})$.

B. Antidictionary

A string $\boldsymbol{v}_1^k$ $(k \geq 1) \in \mathcal{X}^*$ with the following three properties

$$\boldsymbol{v}_1^k \notin \mathcal{D}(\boldsymbol{x}) \tag{6}$$

$$\boldsymbol{v}_1^{k-1} \in \mathcal{D}(\boldsymbol{x}) \tag{7}$$

$$\boldsymbol{v}_2^k \in \mathcal{D}(\boldsymbol{x}) \tag{8}$$

is called a *Minimal Forbidden Word (MFW)* of $\boldsymbol{x}$. An antidictionary $\mathcal{A}(\boldsymbol{x})$ is the set of all the MFWs of $\boldsymbol{x}$. For convenience, we define $\mathcal{D}(\lambda) = \{\lambda\}$ and $\mathcal{A}(\lambda) = \mathcal{X}$. Let $\mathcal{A}_m(\boldsymbol{x})$ be a subset of $\mathcal{A}(\boldsymbol{x})$ in which the length of an element is shorter than or equal to m. For example, $\mathcal{A}(\boldsymbol{x})$ for $\boldsymbol{x} = 01000001$ is given as

$$\mathcal{A}(\boldsymbol{x}) = \{11, 101, 0010, 1001, 10001, 000000, 100001\}. \tag{9}$$

Moreover, for $\boldsymbol{x}^{\langle 2 \rangle} = 0100000101000001$,

$$\mathcal{A}_8(\boldsymbol{x}^{\langle 2 \rangle}) = \{11, 1001, 00100, 10001, 10101, 000000, 100001\}. \tag{10}$$

C. Core

For $\boldsymbol{v} = a\boldsymbol{w}b \in \mathcal{X}^+$ and $a, b \in \mathcal{X}$, $c(\boldsymbol{v}) = \boldsymbol{w} \in \mathcal{X}^*$ is called *core* of $\boldsymbol{v}$. For $\boldsymbol{t} \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$ and $\boldsymbol{u} = c(\boldsymbol{t})$, the set of all $\boldsymbol{u}$ with the following properties

$$0\boldsymbol{u}, 1\boldsymbol{u}, \boldsymbol{u}0, \boldsymbol{u}1 \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle}) \tag{11}$$

is represented by $\mathcal{C}(\boldsymbol{x})$. For example, $\boldsymbol{x} = 01000001$,

$$\mathcal{C}(\boldsymbol{x}) = \{\lambda, 0, 00, 000, 010, 0000\}. \tag{12}$$

D. Antidictionary Tree and Automaton

For a given $\mathcal{A}_n(\boldsymbol{x}^{\langle 2 \rangle})$, an *antidictionary tree (AD-tree)* $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$ is a tree structure that stores all the elements of $\mathcal{A}_n(\boldsymbol{x}^{\langle 2 \rangle})$ [1]. Fig. 1 shows $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$ for $\boldsymbol{x} = 01000001$, where $\mathcal{A}_8(\boldsymbol{x}^{\langle 2 \rangle})$ is shown in (10). In Fig. 1, solid circles and

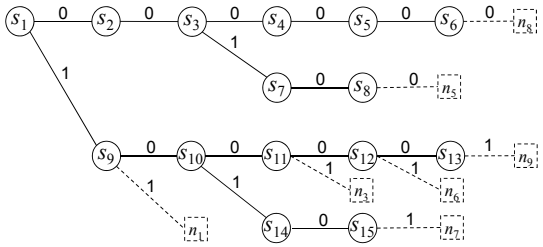

Fig. 1. AD-tree $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$ for $\boldsymbol{x} = 01000001$.

dotted squares respectively represent internal nodes and external nodes called *leaves*. The solid and dotted lines respectively represent edges between internal nodes and between internal nodes and leaves.

In a tree structure, a string associated with a path from the root ρ to a node p in the tree is denoted as $\boldsymbol{w}(p)$. Then we define that $\boldsymbol{w}(\rho)$ is λ. The string $\boldsymbol{w}(p)$ is called *path-string* of p. On the other hand, for a string $\boldsymbol{u}$, let $l(\boldsymbol{u})$ be the node p such as $\boldsymbol{w}(p) = \boldsymbol{u}$, and $l(\boldsymbol{u})$ be called *locus* of $\boldsymbol{u}$. For example, in Fig. 1, $\boldsymbol{w}(s_8)$ is 0010 and $l(1010)$ is s_{15}; s_1 is the root ρ.

An *antidictionary automaton (AD-automaton)* $\mathbb{A}(\boldsymbol{x}^{\langle 2 \rangle})$ is a deterministic automaton constructed from $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$ by means of the L-automaton algorithm shown in [1], [2]. An AD-automaton $\mathbb{A}(\boldsymbol{x}^{\langle 2 \rangle})$ accepts all strings that contain no strings of $\mathcal{A}_n(\boldsymbol{x}^{\langle 2 \rangle})$ as their substrings.

A state of $\mathbb{A}(\boldsymbol{x}^{\langle 2 \rangle})$, corresponding to an internal node of $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$, has two outgoing edges. On the other hand, a state corresponding to a leaf, called *sink*, has no outgoing edge. Edges are defined in the following manner: for each internal node p and $a \in \mathcal{X}$,

(i) if $l(\boldsymbol{w}(p)a)$ exists in $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$, then the edge labeled a from $l(\boldsymbol{w}(p))$ terminates at $l(\boldsymbol{w}(p)a)$.

(ii) if $l(\boldsymbol{w}(p)a)$ does not exist in $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$, then the edge labeled a from $l(\boldsymbol{w}(p))$ terminates at $l(\boldsymbol{v})$, where $\boldsymbol{v}$ is the longest suffix of $\boldsymbol{w}(p)a$ and $\boldsymbol{v} = \boldsymbol{w}(q)$ with node q of $\mathbb{T}_A(\boldsymbol{x}^{\langle 2 \rangle})$.

Fig. 2 shows $\mathbb{A}(\boldsymbol{x}^{\langle 2 \rangle})$ for $\boldsymbol{x} = 01000001$, where $\mathcal{A}_8(\boldsymbol{x}^{\langle 2 \rangle})$ is shown in (10). In Fig. 2, dotted triangles and edges respectively represent sinks and edges to sinks.

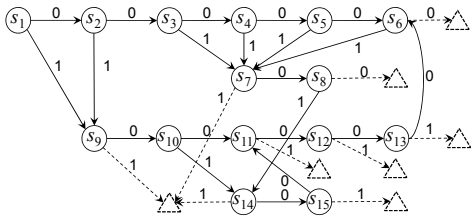

Fig. 2. AD-automaton $\mathbb{A}(\boldsymbol{x}^{\langle 2 \rangle})$ for $\boldsymbol{x} = 01000001$.

E. Compacted Substring Tree and Automaton

For a given $\mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$, a *compacted substring tree (CS-tree)* $\mathbb{T}_C(\boldsymbol{x})$ is a tree structure that stores all the strings $0\boldsymbol{u}$ and $1\boldsymbol{u}$ such that both $0\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$ and $1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$. Fig. 3 shows $\mathbb{T}_C(\boldsymbol{x})$ for $\boldsymbol{x} = 01000001$.

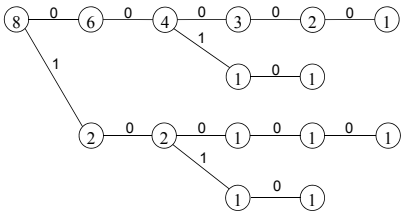

Fig. 3. CS-tree $\mathbb{T}_C(\boldsymbol{x})$ for $\boldsymbol{x} = 01000001$.

For a node p of $\mathbb{T}_C(\boldsymbol{x})$,

$$N(\boldsymbol{w}(p)) = |\{\boldsymbol{u} \,|\, \boldsymbol{w}(p) \in \mathcal{P}(\boldsymbol{u}), \boldsymbol{u} \in \mathcal{W}_n(\boldsymbol{x})\}|. \tag{13}$$

In Fig. 1, a number written in a node p represents $N(\boldsymbol{w}(p))$.

As for $\mathbb{T}_C(\boldsymbol{x})$, Theorem A and Theorem B hold [9].

Theorem A (Theorem 1 [9])**.** *For $\mathbb{T}_C(\boldsymbol{x})$, there exists a node $l(0\boldsymbol{u})$ if and only if $0\boldsymbol{u}, 1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. Similarly, there exists a node $l(1\boldsymbol{u})$ if and only if $0\boldsymbol{u}, 1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$.*

Theorem B (Theorem 2 [9])**.** *For $\mathbb{T}_C(\boldsymbol{x})$, a subtree having $l(0)$ as the root and a subtree having $l(1)$ as the root are isomorphic, both of which have $n-1$ nodes.*

A *compacted substring automaton (CS-automaton)* $\mathbb{C}(\boldsymbol{x})$ is a deterministic automaton that is constructed from $\mathbb{T}_C(\boldsymbol{x})$. A CS-automaton $\mathbb{C}(\boldsymbol{x})$ accepts all the strings in $\mathcal{D}(\boldsymbol{x}^{\langle\infty\rangle})$. A CS-automaton was called CST in an earlier report [6].

A state of $\mathbb{C}(\boldsymbol{x})$, corresponding to a node of $\mathbb{T}_C(\boldsymbol{x})$, has one or two edges. Edges are defined in the following manner: for each node p and $a \in \mathcal{X}$,

(i) if $l(\boldsymbol{w}(p)a)$ exists in $\mathbb{T}_C(\boldsymbol{x})$, then the edge labeled a from $l(\boldsymbol{w}(p))$ terminates at $l(\boldsymbol{w}(p)a)$.

(ii) if $l(\boldsymbol{w}(p)a)$ does not exist in $\mathbb{T}_C(\boldsymbol{x})$ and $\boldsymbol{w}(p)a \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$, then the edge labeled a from $l(\boldsymbol{w}(p))$ terminates at $l(\boldsymbol{u})$, where $\boldsymbol{u}$ is the longest suffix of $\boldsymbol{w}(p)a$ and $\boldsymbol{u}=\boldsymbol{w}(q)$ with node q of $\mathbb{T}_C(\boldsymbol{x})$.

In [6], an edge of (i) is called *forward edge* and an edge of (ii) is called *backward edge*. Fig. 4 shows $\mathbb{C}(\boldsymbol{x})$ for $\boldsymbol{x} = 01000001$.

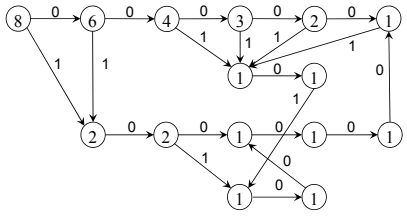

Fig. 4. CS-automaton $\mathbb{C}(\boldsymbol{x})$ for $\boldsymbol{x} = 01000001$.

III. Reviews of Antidictionary and CSE Codings

A. Antidictionary Coding

In a static antidictionary coding, an AD-automaton is used as an encoder, and we use $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$ to compress an input string $\boldsymbol{x}$ in this paper. From the initial state s_1, the transitions with $\boldsymbol{x}$ are implemented. In the transitions, a symbol is output in a state having two outgoing edges to non-sinks, while no symbol is output in a state having one outgoing edge to a sink. In a static encoding, a transition to sink never occurs since the transition corresponds to an occurrence of an MFW. Therefore, the transition from a state having only one outgoing edge to non-sink is predictable.

The static antidictionary coding outputs the following triples

$$(n, \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), \gamma(\boldsymbol{x})) \tag{14}$$

where $\gamma(\boldsymbol{x})$ represents a string concatenating output symbols [2]. As an example, for $\boldsymbol{x} = 01000001$ and $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$ shown in Fig. 2, $(8, \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), 010)$ is output.

B. CSE Coding

In the CSE coding, a CS-automaton is used as an encoder. In the decoding, $\mathbb{C}(\boldsymbol{x})$ is constructed from $\Phi(\boldsymbol{x})$,

$$\Phi(\boldsymbol{x}) = \phi(N(0))\phi(N(0\boldsymbol{w}_1 0)) \dots \phi(N(0\boldsymbol{w}_m 0)) \tag{15}$$

where $\mathcal{C}(\boldsymbol{x}) = \{\boldsymbol{w}_1, \dots, \boldsymbol{w}_m\}$ and $\phi(i)$ is a binary representation of i. The CSE outputs the following triples

$$(n, \Phi(\boldsymbol{x}), \text{rank}(\boldsymbol{x})) \tag{16}$$

where rank($\boldsymbol{x}$) represents the rank of $\boldsymbol{x}$ in the lexicographical order of $\boldsymbol{x}$ in $\mathcal{W}_n(\boldsymbol{x})$ [6]. Note that the set of all the output strings of length n from $\mathbb{C}(\boldsymbol{x})$ is equal to $\mathcal{W}_n(\boldsymbol{x})$. As an example, for $\boldsymbol{x} = 01000001$, (12), and Fig. 3, $(8, \phi(6)\phi(4)\phi(3)\phi(2)\phi(1)\phi(0)\phi(0), 4)$ is output.

IV. Relationship between AD-automaton and CS-automaton

In this section, we will prove the following theorem.

Theorem 1. *For a given $\boldsymbol{x}$, $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$, without all the sinks and the edges to the sinks, and $\mathbb{C}(\boldsymbol{x})$ are isomorphic.*

From the constructions of an AD-automaton and of a CS-automaton, Theorem 1 is a corollary of Proposition 1.

Proposition 1. *For a given $\boldsymbol{x}$, $\mathbb{T}_A(\boldsymbol{x}^{\langle 2\rangle})$, without all the leaves and the edges to the leaves, and $\mathbb{T}_C(\boldsymbol{x})$ are isomorphic.*

To prove Proposition 1, we first give four lemmas.

Lemma 1. *For $\boldsymbol{w} \in \mathcal{C}(\boldsymbol{x})$, $|\boldsymbol{w}| \le n-2$.*

Proof: For $\boldsymbol{w} \in \mathcal{C}(\boldsymbol{x})$, $0\boldsymbol{w} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $1\boldsymbol{w} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ hold. From Theorem A, both $0\boldsymbol{w}$ and $1\boldsymbol{w}$ are path-strings in $\mathbb{T}_C(\boldsymbol{x})$. From Theorem B, $|0\boldsymbol{w}| \le n-1$ (resp. $1\boldsymbol{w}$) since there are just $n-2$ edges in a subtree whose root is $l(0)$ (resp. $l(1)$). Therefore, $|\boldsymbol{w}| \le n-2$. ■

Lemma 2. *For $\boldsymbol{u} \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$, $c(\boldsymbol{u}) \in \mathcal{C}(\boldsymbol{x})$.*

Proof: Since $n \ge 2$ and Condition 1 hold, $2 \le |\boldsymbol{u}| \le n$ holds. Therefore, for $a, b \in \mathcal{X}$ and $\boldsymbol{w} \in \mathcal{X}^*$, $\boldsymbol{u}$ is written as $a\boldsymbol{w}b$. From (7) and (8),

$$a\boldsymbol{w}, \boldsymbol{w}b \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}). \tag{17}$$

Since $|\boldsymbol{w}b| \le n-1$, there exists symbol c such that $c\boldsymbol{w}b \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. However, since $a\boldsymbol{w}b \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$, $c\boldsymbol{w}b = \bar{a}\boldsymbol{w}b$ holds. Therefore,

$$\bar{a}\boldsymbol{w} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}). \tag{18}$$

Similarly, since $|a\boldsymbol{w}| \le n-1$, there exists symbol d such that $a\boldsymbol{w}d \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. However, since $a\boldsymbol{w}b \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$, $a\boldsymbol{w}d = a\boldsymbol{w}\bar{b}$ holds. Therefore,

$$\boldsymbol{w}\bar{b} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}). \tag{19}$$

From (17), (18), and (19), $\boldsymbol{w} = c(\boldsymbol{u}) \in \mathcal{C}(\boldsymbol{x})$. ■

Lemma 3. *For $\boldsymbol{w} \in \mathcal{C}(\boldsymbol{x})$, there exists $a \in \mathcal{X}$ and $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}^+$ such that $a\boldsymbol{w}\boldsymbol{u} \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$ and $\bar{a}\boldsymbol{w}\boldsymbol{v} \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$.*

Proof: From Lemma 1, $|\boldsymbol{w}| \le n-2$. Since $\boldsymbol{w} \in \mathcal{C}(\boldsymbol{x})$, there exists $\boldsymbol{t} \in \mathcal{X}^+$ such that $a\boldsymbol{w}\boldsymbol{t} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $|a\boldsymbol{w}\boldsymbol{t}| =$

n. From Theorem B, $\bar{a}\boldsymbol{wt} \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ since $|\boldsymbol{wt}| = n-1$. Therefore, there exists $k \geq 0$ such that $\pi^k(\bar{a}\boldsymbol{wt}) \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $\pi^{k+1}(\bar{a}\boldsymbol{wt}) \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ since $\bar{a}\boldsymbol{w} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. Let $\boldsymbol{v}$ be $\pi^k(\boldsymbol{t})$. Since $a\boldsymbol{wv} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$,

$$\bar{a}\boldsymbol{wv} \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{20}$$

$$\pi(\bar{a}\boldsymbol{wv}) \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{21}$$

$$\sigma(\bar{a}\boldsymbol{wv}) = \boldsymbol{wv} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{22}$$

hold. From (21) and Theorem B, $|\pi(\bar{a}\boldsymbol{wv})| = |\boldsymbol{wv}| \leq n-1$. Since (20), (21), (22), and $|\boldsymbol{wv}| \leq n-1$, $\bar{a}\boldsymbol{wv} \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$.

Similarly, there exists $\boldsymbol{s} \in \mathcal{X}^+$ such that $\bar{a}\boldsymbol{ws} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $|\bar{a}\boldsymbol{ws}| = n$. From Theorem B, $a\boldsymbol{ws} \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ since $|\boldsymbol{ws}| = n-1$. Therefore, there exists $l \geq 0$ such that $\pi^l(a\boldsymbol{ws}) \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $\pi^{l+1}(a\boldsymbol{ws}) \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ since $a\boldsymbol{w} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. Letting $\boldsymbol{u}$ be $\pi^l(\boldsymbol{s})$, then since $\bar{a}\boldsymbol{wu} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$,

$$a\boldsymbol{wu} \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{23}$$

$$\pi(a\boldsymbol{wu}) \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{24}$$

$$\sigma(a\boldsymbol{wu}) = \boldsymbol{wu} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle}) \tag{25}$$

hold. From (24) and Theorem B, $|\pi(\bar{a}\boldsymbol{wu})| = |\boldsymbol{wu}| \leq n-1$. Since (23), (24), (25), and $|\boldsymbol{wu}| \leq n-1$, $a\boldsymbol{wu} \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$. ■

Lemma 4. *Both $0\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ hold if and only if $\boldsymbol{u} \in \mathcal{P}(\boldsymbol{v})$ where $\boldsymbol{v} \in \mathcal{C}(\boldsymbol{x})$.*

Proof: For $\boldsymbol{v} \in \mathcal{C}(\boldsymbol{x})$, $0\boldsymbol{v}, 1\boldsymbol{v} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. Therefore, for $\boldsymbol{u} \in \mathcal{P}(\boldsymbol{v})$, $0\boldsymbol{u}, 1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$.

For $0\boldsymbol{u}, 1\boldsymbol{u} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$, from Theorem B, there exists $\boldsymbol{z} \in \mathcal{X}^*$ and $a \in \mathcal{X}$ such that $0\boldsymbol{uz}a, 1\boldsymbol{uz}\bar{a} \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ and $|\boldsymbol{uz}| \leq n-2$. Let $\boldsymbol{v}$ be $\boldsymbol{uz}$, and $\boldsymbol{v} = \boldsymbol{uz} \in \mathcal{C}(\boldsymbol{x})$ since $0\boldsymbol{uz}, 1\boldsymbol{uz}, \boldsymbol{uz}0, \boldsymbol{uz}1 \in \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$. ■

(Proof of Proposition 1): From Theorem A and Lemma 4, the set of all the path-strings of $\mathbb{T}_C(\boldsymbol{x})$ is given as

$$\{\boldsymbol{u} \,|\, \boldsymbol{u} \in \mathcal{P}(a\boldsymbol{v}), \boldsymbol{v} \in \mathcal{C}(\boldsymbol{x}), a \in \mathcal{X}\}. \tag{26}$$

A set of all path-strings of $\mathbb{T}_A(\boldsymbol{x})$ without all leaves and edges to leaves is given as

$$\{\boldsymbol{w} \,|\, \boldsymbol{w} \in \mathcal{P}(a\boldsymbol{z}), a\boldsymbol{z}b \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), b \in \mathcal{X}\}. \tag{27}$$

For any node p of $\mathbb{T}_C(\boldsymbol{x})$, from Lemma 3 and Lemma 4,

$$\boldsymbol{w}(p) \in \{\boldsymbol{w} \,|\, \boldsymbol{w} \in \mathcal{P}(a\boldsymbol{z}), a\boldsymbol{z}b \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), b \in \mathcal{X}\}. \tag{28}$$

From (26) and (28),

$$\begin{aligned}&\{\boldsymbol{u} \,|\, \boldsymbol{u} \in \mathcal{P}(a\boldsymbol{v}), \boldsymbol{v} \in \mathcal{C}(\boldsymbol{x}), a \in \mathcal{X}\}\\ &\subset \{\boldsymbol{w} \,|\, \boldsymbol{w} \in \mathcal{P}(a\boldsymbol{z}), a\boldsymbol{z}b \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), b \in \mathcal{X}\}.\end{aligned} \tag{29}$$

For any node q of $\mathbb{T}_A(\boldsymbol{x}^{\langle 2\rangle})$ without the leaves and the edges to the leaves, from Lemma 2 and Lemma 4,

$$\boldsymbol{w}(q) \in \{\boldsymbol{u} \,|\, \boldsymbol{u} \in \mathcal{P}(a\boldsymbol{v}), \boldsymbol{v} \in \mathcal{C}(\boldsymbol{x}), a \in \mathcal{X}\}. \tag{30}$$

From (27) and (30),

$$\begin{aligned}&\{\boldsymbol{w} \,|\, \boldsymbol{w} \in \mathcal{P}(a\boldsymbol{z}), a\boldsymbol{z}b \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), b \in \mathcal{X}\}\\ &\subset \{\boldsymbol{u} \,|\, \boldsymbol{u} \in \mathcal{P}(a\boldsymbol{v}), \boldsymbol{v} \in \mathcal{C}(\boldsymbol{x}), a \in \mathcal{X}\}.\end{aligned} \tag{31}$$

From (31) and (29),

$$\begin{aligned}&\{\boldsymbol{u} \,|\, \boldsymbol{u} \in \mathcal{P}(a\boldsymbol{v}), \boldsymbol{v} \in \mathcal{C}(\boldsymbol{x}), a \in \mathcal{X}\}\\ &= \{\boldsymbol{w} \,|\, \boldsymbol{w} \in \mathcal{P}(a\boldsymbol{z}), a\boldsymbol{z}b \in \mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle}), b \in \mathcal{X}\}.\end{aligned} \tag{32}$$

From (32), the set of all the path-strings of $\mathbb{T}_C(\boldsymbol{x})$ is the same as the set of all the path-strings of $\mathbb{T}_A(\boldsymbol{x}^{\langle 2\rangle})$ without all the leaves and the edges to the leaves. ■

From Proposition 1, for a node p of $\mathbb{T}_C(\boldsymbol{x})$ and a node q of $\mathbb{T}_A(\boldsymbol{x}^{\langle 2\rangle})$ without all the leaves and the edges to the leaves, a bijective function f such as $f : p \to q$ is definable. For nodes s, t and $a \in \mathcal{X}$ in $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$ and $\mathbb{C}(\boldsymbol{x})$, such that the edge from s to t is labeled by symbol a, let (s, t) be the symbol a.

(Proof of Theorem 1): For the labelling symbol $a = (p, q)$ where p is a not sink and q is a sink, $\boldsymbol{w}(p)a \notin \mathcal{D}(\boldsymbol{x}^{\langle 2\rangle})$ since $\boldsymbol{w}(p)a$ has an MFW in $\mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$ as its suffix. Therefore, $N(\boldsymbol{w}(p)a) = 0$.

From both the constructions of $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$ and of $\mathbb{C}(\boldsymbol{x})$ and Proposition 1, for p and q of $\mathcal{C}(\boldsymbol{x})$ such that (p, q) exists, there exists a bijective function f such that $(p, q) = (f(p), f(q))$ where $f(p)$ and $f(q)$ are the nodes of $\mathbb{A}(\boldsymbol{x}^{\langle 2\rangle})$ without all the sinks and the edges to the sinks. ■

V. Proposed Algorithm

The proposed algorithm uses $\Phi(\boldsymbol{x})$ to represent $\mathcal{A}_n(\boldsymbol{x}^{\langle 2\rangle})$ in (14). The proposed algorithm outputs

$$(0, n, \Phi(\boldsymbol{x}), \gamma(\boldsymbol{x})) \quad (\boldsymbol{x} \in \mathcal{B}_n) \tag{33}$$

$$(1, n, c, \Phi(\boldsymbol{x}_1^c), \gamma(\boldsymbol{x}_1^c)) \quad (\boldsymbol{x} \notin \mathcal{B}_n) \tag{34}$$

as a codeword for a given string $\boldsymbol{x}$, and c is the shortest length such that $\boldsymbol{x} = \boldsymbol{x}_1^c \boldsymbol{x}_1^c \dots \boldsymbol{x}_1^c$ and $c < n$. In other words, $\boldsymbol{x}_1^c \in \mathcal{B}_n$ and c is a divisor of n.

Let $\mathbf{X}$ be a stationary ergodic source, and let $H(\mathbf{X})$ be the entropy rate of $\mathbf{X}$ and $H(\mathbf{X}) < \infty$. Then $\mathbf{X}_1^n$ is a sequence of random variables $X_1 \dots X_n$ on $\mathbf{X}$. The recurrence-time R_l is defined as $R_l = \min\{i > 0 \,|\, \mathbf{X}_1^l = \mathbf{X}_{1+i}^{l+i}\}$.

Let ℓ_n be the codeword length per symbol of the proposed algorithm for a random string of length n. We will prove Theorem 2.

Theorem 2. *For any stationary ergodic source $\mathbf{X}$, ℓ_n converges to $H(\mathbf{X})$ in probability as $n \to \infty$.*

To prove Theorem 2, we use Lemma 5 below and the following theorem:

Theorem C (Theorem 2 [7])**.** *For any stationary ergodic source $\mathbf{X}$, $\lim_{n\to\infty} \frac{L_{CSE}(\mathbf{X}_1^n)}{n} = H(\mathbf{X})$ with probability one where $L_{CSE}(\mathbf{X}_1^n)/n$ is the model entropy per bit of the CSE.*

We convince the Theorem C, however, the case $\boldsymbol{x} \notin \mathcal{B}_n$ is not considered appropriately in the original proof. Hence, we first prepare a proposition.

Proposition 2. *If the entropy rate $H(\mathbf{X})$ of $\boldsymbol{X}$ is positive, then,*

$$\lim_{n\to\infty} P(\mathbf{X}_1^n \notin \mathcal{B}_n) = 0.$$

Proof: For $\boldsymbol{x} \in \mathcal{B}_n$, there exists a positive integer $k < n$, such that k is a divisor of n, which satisfies $\boldsymbol{x}_1^k = \boldsymbol{x}_{n-k+1}^n$

and $\boldsymbol{x}_1^{n-k} = \boldsymbol{x}_{k+1}^n$. In other words, $\boldsymbol{x} = \boldsymbol{x}_1^k \boldsymbol{x}_1^k \dots \boldsymbol{x}_1^k$. Since $1 \le k \le \lfloor \frac{n}{2} \rfloor$, $\lceil \frac{n}{2} \rceil \le n-k < n$ holds. Then,

$$P(\mathbf{X}_1^n \notin \mathcal{B}_n) \le P\left(R_{\lceil n/2 \rceil} \le \left\lfloor \frac{n}{2} \right\rfloor\right) \tag{35}$$

$$\le P\left(R_{n/2} \le \frac{n}{2}\right) \tag{36}$$

$$= P\left(\frac{\log_2 R_{n/2}}{n/2} \le \frac{\log_2 n/2}{n/2}\right) \tag{37}$$

Theorem II.5.1 [10] shows $\lim_{m\to\infty} \frac{\log_2 R_m}{m} = H(\mathbf{X})$ with probability 1. Hence, for arbitrary $\varepsilon > 0$ and $\delta > 0$, there exists n_δ such that for any $m \ge n_\delta$

$$P\left(\bigcup_{k=m}^{\infty} \left\{ \left| \frac{1}{k} \log_2 R_k - H(\boldsymbol{X}) \right| \ge \varepsilon \right\}\right) \le \delta,$$

which implies

$$P\left(\left\{ \left| \frac{1}{m} \log_2 R_m - H(\boldsymbol{X}) \right| \ge \varepsilon \right\}\right) \le \delta.$$

Since $H(\boldsymbol{X}) < \infty$, it holds $\frac{\log n/2}{n/2} \le H(\boldsymbol{X}) - \varepsilon$ if ε is small and n is sufficiently large. Therefore, letting $m = n/2$, we have

$$P(\mathbf{X}_1^n \notin \mathcal{B}_n) \le P\left(\frac{\log_2 R_m}{m} \le H(\boldsymbol{X}) - \varepsilon\right) \le \delta.$$

Since δ is arbitrarily small, the statement of the proposition holds. ∎

Lemma 5. *$|\gamma(\boldsymbol{x})| \le |\boldsymbol{w}_m| + 1$ holds where $\boldsymbol{w}_m$ is the longest string in $\{\boldsymbol{w} | \boldsymbol{w}0, \boldsymbol{w}1 \in \mathcal{D}(\boldsymbol{x}^{\langle\infty\rangle})\}$.*

Proof: We assume that $|\gamma(\boldsymbol{x})| > |\boldsymbol{w}_m| + 1$. There exists $\boldsymbol{u}a \in \mathcal{P}(\boldsymbol{x})$ such that $|\gamma(\boldsymbol{u})| = |\boldsymbol{w}_m| + 1$ and $|\gamma(\boldsymbol{u}a)| = |\boldsymbol{w}_m| + 2$ where $a \in \mathcal{X}$. Since the state transition with $\boldsymbol{u}$ from the initial state of $\mathbb{C}(\boldsymbol{x})$ has two outgoing edges, $\boldsymbol{u}\bar{a} \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$. Therefore, $\boldsymbol{u}0, \boldsymbol{u}1 \in \mathcal{D}(\boldsymbol{x}^{\langle 2 \rangle})$ so that $|\boldsymbol{u}| > |\boldsymbol{w}_m|$. It contradicts of maximality of $\boldsymbol{w}_m$. Hence, $|\gamma(\boldsymbol{x})| \le |\boldsymbol{w}_m| + 1$. ∎

Now we give a proof of Theorem 2.

(Proof of Theorem 2): For $\mathbf{X}$, we consider $H(\mathbf{X}) > 0$ and $H(\mathbf{X}) = 0$. First, we show the proof in case of $H(\mathbf{X}) > 0$. For any $\delta > 0$,

$$\begin{aligned} &P(|\ell_n - H(\mathbf{X})| > \delta) \\ &= P(|\ell_n - H(\mathbf{X})| > \delta \,|\, \mathbf{X} \in \mathcal{B}_n) P(\mathbf{X} \in \mathcal{B}_n) \\ &+ P(|\ell_n - H(\mathbf{X})| > \delta \,|\, \mathbf{X} \notin \mathcal{B}_n) P(\mathbf{X} \notin \mathcal{B}_n) \qquad (38) \\ &\le P(|\ell_n - H(\mathbf{X})| > \delta \,|\, \mathbf{X} \in \mathcal{B}_n) + P(\mathbf{X} \notin \mathcal{B}_n) \qquad (39) \end{aligned}$$

From Proposition 2, the second term of right-hand side in (39) converges to 0 from as $n \to \infty$.

Next, we consider the first term of right-hand side in (39). From Theorem C, for $\boldsymbol{x} \in \mathcal{B}_n$, the codeword length in (16) per symbol of the CSE converges to $H(\mathbf{X})$ with probability one. From (16) and (33), the proposed algorithm outputs a pair $(0, \gamma(\boldsymbol{x}))$ instead of rank($\boldsymbol{x}$) against the codeword of the CSE. Therefore, we will prove $\frac{1+|\gamma(\boldsymbol{x})|}{n}$ converges to 0 in probability as n goes to infinity, so that Theorem 2 holds in case of $\boldsymbol{x} \in \mathcal{B}_n$.

From Lemma 5, $|\gamma(\boldsymbol{x})| \le |\boldsymbol{w}_m| + 1$. Then $|\boldsymbol{w}_m| + 1$ is the height of the suffix tree of $\boldsymbol{x}^{\langle\infty\rangle}$. Moreover, from Lemma 2 [7], $\boldsymbol{w}_m \in \mathcal{C}(\boldsymbol{x})$. Hence, $|\boldsymbol{w}_m| + 1$ is also the height of $\mathbb{T}_C(\boldsymbol{x})$.

From Theorem II.5.4 [10], the average height of $\mathbb{T}_C(\mathbf{X})$ is given by $c_1 \log_2 n$ where c_1 is a positive constant with probability one. Therefore, $|\gamma(\boldsymbol{x})| = c_1 \log_2 n + 1$ holds with probability one. Then, $\lim_{n\to\infty} \frac{|\gamma(\boldsymbol{x})|}{n} = 0$ in probability. Therefore, ℓ_n converges to $H(\mathbf{X})$ as n goes to infinity since Theorem C holds and $\frac{1+|\gamma(\boldsymbol{x})|}{n}$ converges to 0 in probability as n goes to infinity. Theorem 2 holds in case of $H(\mathbf{X}) > 0$.

We next consider in case of $H(\mathbf{X}) = 0$. For $\boldsymbol{x}$ on $\mathbf{X}$ with $H(\mathbf{X}) = 0$ and sufficient large n, $\boldsymbol{x}$ can be written by a substring of $\boldsymbol{z}^{\langle\infty\rangle}$ where $\boldsymbol{z}$ is a string of constant length c. Therefore, $|\gamma(\boldsymbol{x})| \le c$ since the height of $\mathbb{T}_C(\boldsymbol{x})$ is shorter than and equal to c. Hence, $\lim_{n\to\infty} \frac{|\gamma(\boldsymbol{x})|}{n} = 0$ holds.

Therefore, in case of $\boldsymbol{x} \in \mathcal{B}_n$, ℓ_n converges to 0 ($= H(\mathbf{X})$) as n goes to infinity since Theorem C holds and $\frac{1+|\gamma(\boldsymbol{x})|}{n}$ converges to 0 as n goes to infinity. As for $\boldsymbol{x} \notin \mathcal{B}_n$, the length of a part of codeword $(1, c, \Phi(\boldsymbol{x}_1^c), \gamma(\boldsymbol{x}_1^c))$ is a positive constant since c is a constant. Then length of binary representation of n is $O(\log n)$. Hence, in case of $\boldsymbol{x} \notin \mathcal{B}_n$, ℓ_n also converges to 0 ($= H(\mathbf{X})$) as n goes to infinity. Therefore, Theorem 2 also holds in case of $H(\mathbf{X}) = 0$. ∎

VI. Conclusion

For a given binary string satisfying Condition 1, the results presented herein prove that an antidictionary automaton, without all the sinks and the edges to the sinks, and a compacted substring automaton are isomorphic. Then we proposed a new antidictionary coding which uses a representation of compacted substring automaton as antidictionary. Moreover, we proved the asymptotic optimality of the proposed algorithm for a stationary ergodic source.

Acknowledgment

The authors would like to thank Dr. Danny Dubé for helpful discussions with respect to construction of the CS-automaton from an input string. This research is partly supported by Grant-in-Aid for Scientific Research(C):24500110.

References

[1] M. Crochemore, F. Mignosi and A. Restivo, "Automata and forbidden words," *Inform. Processing Lett.*, 67(3), pp.111–117, 1998(8).

[2] M. Crochemore, F. Mignosi, A. Restivo and S. Salemi, "Data compression using antidictionaries," *Proc. IEEE*, 88(11), pp.1756–1768, 2000(11).

[3] M. Fiala and J. Holub, "DCA using suffix arrays," *Proc. DCC2008*, pp.516, 2008(5).

[4] T. Ota and H. Morita, "On the adaptive antidictionary code using minimal forbidden words with constant lengths," *Proc. ISITA2010*, pp.72-77, 2010(10).

[5] T. Ota and H. Morita, "Asymptotic optimality of antidictionary codes," *Proc. ISIT2010*, pp.101-105, 2010(6).

[6] D. Dubé and V. Beaudoin, "Lossless data compression via substring enumeration," *Proc. DCC2010*, pp.229-238, 2010(3).

[7] H. Yokoo, "Asymptotic optimal lossless compression via the CSE technique," *Proc. CCP2011*, pp.11-18, 2011(6).

[8] M. Lothaire, "Applied combinatorics on words," *Cambridge*, p.4, 2005.

[9] D. Dubé and H. Yokoo, "The universality and linearity of compression by substring enumeration," *Proc. ISIT2011*, pp.1619-1623, 2011(8).

[10] P.C. Shields, "The ergodic theory of discrete sample paths," *A.M.S.*, pp.154-159, 1996.