Optimality of Huffman Code in the Class of 1-Bit Delay Decodable Codes

Kengo Hashimoto, *Graduate Student Member, IEEE*, and Ken-ichi Iwata, *Member, IEEE*

***Abstract*—For a given independent and identically distributed (i.i.d.) source, Huffman code achieves the optimal average codeword length in the class of instantaneous codes with a single code table. However, it is known that there exist time-variant encoders, which achieve a shorter average codeword length than the Huffman code, using multiple code tables and allowing at most k-bit decoding delay for $k = 2, 3, 4, \ldots$. On the other hand, it is not known whether there exists a 1-bit delay decodable code, which achieves a shorter average length than the Huffman code. This paper proves that for a given i.i.d. source, a Huffman code achieves the optimal average codeword length in the class of 1-bit delay decodable codes with a finite number of code tables.**

***Index Terms*—Data compression, source coding, Huffman code, decoding delay.**

I. Introduction

We consider the data compression system shown in Fig. 1. The i.i.d. source outputs a sequence $\boldsymbol{x} = x_1x_2\ldots x_n$ of symbols of the source alphabet $\mathcal{S} = \{s_1, s_2, \ldots, s_\sigma\}$, where n and σ denote the length of $\boldsymbol{x}$ and the alphabet size, respectively. Each source output follows a fixed probability distribution $(\mu(s_1), \mu(s_2), \ldots, \mu(s_\sigma))$, where $\mu(s_i)$ is the probability of occurrence of s_i for $i = 1, 2, \ldots, \sigma$. In this paper, we assume $\sigma \geq 2$. The encoder reads the source sequence $\boldsymbol{x}$ symbol by symbol from the beginning of $\boldsymbol{x}$ and encodes them according to a time-variant code table $f : \mathcal{S} \to \mathcal{C}^*$, where $\mathcal{C}^*$ denotes the set of all sequences of finite length over the coding alphabet $\mathcal{C} := \{0, 1\}$. As the result, $\boldsymbol{x} = x_1x_2\ldots x_n$ is encoded to a binary codeword sequence $f(\boldsymbol{x}) := f(x_1)f(x_2)\ldots f(x_n)$. Then the decoder reads the codeword sequence $f(\boldsymbol{x})$ bit by bit from the beginning of $f(\boldsymbol{x})$ and recovers the original source sequence $\boldsymbol{x}$.

For example, we consider an i.i.d. source with source alphabet $\mathcal{S} = \{\mathrm{a, b, c, d}\}$, a probability distribution $(\mu(\mathrm{a}), \mu(\mathrm{b}), \mu(\mathrm{c}), \mu(\mathrm{d})) = (0.1, 0.2, 0.3, 0.4)$ and the time-invariant code table $f^{(\mathrm{I})}$ in Table I. The *average codeword length* $L(f^{(\mathrm{I})})$ of the code table $f^{(\mathrm{I})}$ is calculated as

$$L\left(f^{(\mathrm{I})}\right) = \sum_{s\in\{\mathrm{a,b,c,d}\}} \mu(s)|f^{(\mathrm{I})}(s)| = 1.9, \qquad (1)$$

Manuscript received 15 April 2022; revised 19 September 2022; accepted 15 December 2022. Date of publication 23 December 2022; date of current version 13 June 2023. This work was supported in part by JSPS KAKENHI under Grant JP18H01436 and Grant JP20K11674, and in part by KIOXIA. *(Corresponding author: Kengo Hashimoto.)*

The authors are with the Graduate School of Engineering, University of Fukui, Fukui 910-8507, Japan (e-mail: khasimot@u-fukui.ac.jp; k-iwata@u-fukui.ac.jp).

Digital Object Identifier 10.1109/JSAIT.2022.3230745

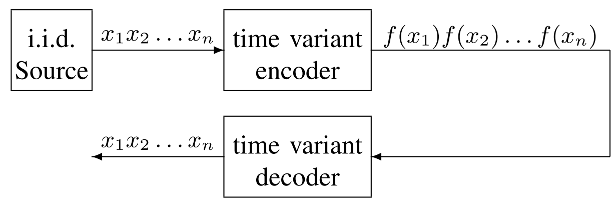

Fig. 1. Diagram of data compression system.

TABLE I
An Example of a Code Table

$s \in \mathcal{S}$	$f^{(\mathrm{I})}$
a	000
b	001
c	01
d	1

where $|\cdot|$ denotes the length of a sequence.

A source sequence $\boldsymbol{x} = x_1x_2x_3x_4x_5 = \mathrm{adbac}$ is encoded to $f^{(\mathrm{I})}(\boldsymbol{x}) = f^{(\mathrm{I})}(\mathrm{a})f^{(\mathrm{I})}(\mathrm{d})f^{(\mathrm{I})}(\mathrm{b})f^{(\mathrm{I})}(\mathrm{a})f^{(\mathrm{I})}(\mathrm{c}) = 000100100001$. The code table $f^{(\mathrm{I})}$ is *prefix-free*; that is, for any two distinct symbols s and s', $f(s)$ is not a prefix of $f(s')$. Therefore, in the decoding process, the decoder can uniquely determine $x_1 = \mathrm{a}$ immediately after reading the prefix $f^{(\mathrm{I})}(\mathrm{a}) = 000$ of $f^{(\mathrm{I})}(\boldsymbol{x})$. Also, the decoder can uniquely determine $x_1x_2 = \mathrm{ad}$ immediately after reading the prefix $f^{(\mathrm{I})}(\mathrm{a})f^{(\mathrm{I})}(\mathrm{d}) = 0001$ of $f^{(\mathrm{I})}(\boldsymbol{x})$. In general, a code table f is called an *instantaneous code* if for any source sequence $\boldsymbol{x} = x_1x_2\ldots x_n$ and its any prefix $x_1x_2\ldots x_l$, the decoder can uniquely recover $x_1x_2\ldots x_l$ immediately after reading the prefix $f(x_1)f(x_2)\ldots f(x_l)$. In fact, the code table $f^{(\mathrm{I})}$ is an instantaneous code.

Huffman [1] proposed an algorithm to construct an instantaneous code that achieves the optimal average codeword length for a given probability distribution $(\mu(s_1), \mu(s_2), \ldots, \mu(s_\sigma))$. A code table constructed by Huffman's algorithm is called a *Huffman code*. Namely, a Huffman code achieves the optimal average codeword length in the class of instantaneous codes. McMillan's theorem [2] implies that Huffman code is optimal also in the class of uniquely decodable codes (with a single code table). The code table $f^{(\mathrm{I})}$ in Table I is a Huffman code for the probability distribution $(\mu(\mathrm{a}), \mu(\mathrm{b}), \mu(\mathrm{c}), \mu(\mathrm{d})) = (0.1, 0.2, 0.3, 0.4)$.

In 2015, Yamamoto, Tsuchihashi, and Honda [3] proposed binary AIFV code that can achieve a shorter average codeword length than the Huffman code. An AIFV code uses a time-variant encoder consisting of two code tables f_0, f_1 and allows

TABLE II
AN EXAMPLE OF AN AIFV CODE $F^{(\mathrm{II})}(f_0, f_1, \tau_0, \tau_1)$

$s \in \mathcal{S}$	f_0	τ_0	f_1	τ_1
a	100	0	1100	0
b	00	0	11	1
c	01	0	01	0
d	1	1	10	0

at most 2-bit delay for decoding. We omit details of the definition of AIFV code here and show an example of an AIFV code $F^{(\mathrm{II})}$ in Table II. In the encoding process of $\boldsymbol{x} = x_1x_2 \dots x_n$ with the AIFV code $F^{(\mathrm{II})}$, the first symbol x_1 is encoded with the code table f_0. For $i = 2, 3, \dots, n$, if x_{i-1} is encoded with the code table f_j, then x_i is encoded with the code table $f_{\tau_j(x_{i-1})}$. For example, a source sequence $\boldsymbol{x} = x_1x_2x_3x_4x_5 = \mathrm{adbac}$ is encoded to $f_0(\mathrm{a})f_0(\mathrm{d})f_1(\mathrm{b})f_1(\mathrm{a})f_0(\mathrm{c}) = 100111110001$.

AIFV code $F^{(\mathrm{II})}$ is not an instantaneous code because the decoder cannot uniquely determine whether $x_1x_2 = \mathrm{ad}$ or not at the time of reading $f_0(\mathrm{a})f_0(\mathrm{d}) = 1001$ (there are still two possibilities, $\boldsymbol{x} = \mathrm{aa}\dots$ and $\boldsymbol{x} = \mathrm{ad}\dots$). However, the decoder can distinguish them by reading the following two bits; that is, the decoder can uniquely determine $x_1x_2 = \mathrm{ad}$ immediately after reading $f_0(\mathrm{a})f_0(\mathrm{d})11 = 100111$. Similarly, when the decoder reads the prefix $f_0(\mathrm{a})f_0(\mathrm{d})f_1(\mathrm{b})f_1(\mathrm{b}) = 10011111$, the decoder cannot uniquely determine whether $x_1x_2x_3x_4 = \mathrm{adbb}$ or not because there are still two possibilities, $\boldsymbol{x} = \mathrm{adbb}\dots$ and $\boldsymbol{x} = \mathrm{adba}\dots$. Also, in this case, the decoder can distinguish them by reading the following two bits, that is, the decoder can uniquely determine $x_1x_2x_3x_4 \neq \mathrm{adbb}$ immediately after reading $f_0(\mathrm{a})f_0(\mathrm{d})f_1(\mathrm{b})f_1(\mathrm{b})00 = 1001111100$ since a 00 cannot follow after encoding b with f_1 in the AIFV code $F^{(\mathrm{II})}$. In general, for the AIFV code $F^{(\mathrm{II})}$, the following condition holds for any source sequence $\boldsymbol{x} = x_1x_2 \dots x_n$: at the time immediately after the decoder reads the prefix $b_1b_2 \dots b_k$ of the codeword sequence $f(\boldsymbol{x})$, if $b_1b_2 \dots b_k = f(x'_1)f(x'_2) \dots f(x'_l)$, then the decoder can uniquely determine whether $x'_1x'_2 \dots x'_l$ is a prefix of the original source sequence $\boldsymbol{x}$ or not by reading the following 2 bits. Namely, $F^{(\mathrm{II})}$ is a 2-*bit delay decodable code* defined formally in Section II-B.

The average codeword length of an AIFV code $F(f_0, f_1, \tau_0, \tau_1)$ is calculated as follows. Let $I_i \in \{0, 1\}$ be the index of the code table used to encode the i-th symbol of the source sequence, for $i = 1, 2, 3, \dots$. Then $I_i, i = 1, 2, 3, \dots$ is a Markov process. Let (π_0, π_1) be the stationary probability of the Markov process, that is,

$$\pi_0 = \frac{Q_{1,0}}{Q_{0,1} + Q_{1,0}}, \quad \pi_1 = \frac{Q_{0,1}}{Q_{0,1} + Q_{1,0}}, \tag{2}$$

where $Q_{i,j}$ is the probability of using f_j immediately after using f_i, for $i, j \in \{0, 1\}$. Then the average codeword length $L(F)$ of an AIFV code F is $L(F) = \pi_0 L_0(F) + \pi_1 L_1(F)$, where $L_0(F)$ (resp. $L_1(F)$) is the average codeword length of the single code table f_0 (resp. f_1). Thus, the average codeword length of the AIFV code $F^{(\mathrm{II})}$ in Table II is

$$\begin{aligned} L\left(F^{(\mathrm{II})}\right) &= \frac{Q_{1,0}}{Q_{0,1} + Q_{1,0}} L_0(F^{(\mathrm{II})}) + \frac{Q_{0,1}}{Q_{0,1} + Q_{1,0}} L_1(F^{(\mathrm{II})}) && (3) \\ &= 1.86666\dots, && (4) \end{aligned}$$

which is shorter than $L(f^{(I)})$ of the Huffman code $f^{(\mathrm{I})}$ in Table I (cf. (1)). We indicate that binary AIFV code achieves the optimal average codeword length in the class of 2-bit delay decodable codes with two code tables [4].

Binary AIFV code is generalized to binary AIFV-m code, which can achieve a shorter average codeword length than binary AIFV code for $m \geq 3$, allowing m code tables and at most m-bit decoding delay [5]. Analyses of the worst-case redundancy of AIFV and AIFV-m codes are studied in the literature [5], [6] for $m = 2, 3, 4, 5$. The papers [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20] propose the code construction and coding method of AIFV and AIFV-m codes. Extensions of AIFV-m codes are proposed in the papers [21], [22].

As stated above, there exist m-bit delay decodable codes with a finite number of code tables better than Huffman code for $m \geq 2$. However, it is not known for the case $m = 1$. In this paper, we prove that there are no 1-bit delay decodable codes with a finite number of code tables, which achieves a shorter average codeword length than the Huffman code. Namely, Huffman code is optimal in the class of 1-bit delay decodable codes with a finite number of code tables.

II. PRELIMINARIES

First, we define some notations as follows. Let $|\mathcal{A}|$ denote the cardinality of a finite set $\mathcal{A}$. Let $\mathcal{A} \times \mathcal{B}$ denote the set of all ordered pairs (a, b), $a \in \mathcal{A}$ and $b \in \mathcal{B}$, that is, $\mathcal{A} \times \mathcal{B} := \{(a, b) : a \in \mathcal{A}, b \in \mathcal{B}\}$. Let $\mathcal{A}^k$ denote the set of all sequences of length k over a set $\mathcal{A}$, and let $\mathcal{A}^*$ denote the set of all sequences of finite length over a set $\mathcal{A}$, that is, $\mathcal{A}^* := \mathcal{A}^0 \cup \mathcal{A}^1 \cup \mathcal{A}^2 \cup \cdots$. The empty sequence is denoted by λ. The length of a sequence $\boldsymbol{x}$ is denoted by $|\boldsymbol{x}|$. In particular, $|\lambda| = 0$. We say $\boldsymbol{x} \preceq \boldsymbol{y}$ if $\boldsymbol{x}$ is a prefix of $\boldsymbol{y}$, that is, there exists a sequence $\boldsymbol{z}$, possibly $\boldsymbol{z} = \lambda$, such that $\boldsymbol{y} = \boldsymbol{x}\boldsymbol{z}$. For a non-empty sequence $\boldsymbol{x} = x_1x_2 \dots x_n$, we define $\mathrm{suf}(\boldsymbol{x}) := x_2x_3 \dots x_n$, that is, $\mathrm{suf}(\boldsymbol{x})$ is the sequence obtained by deleting the first letter x_1 from $\boldsymbol{x}$. Some notations used in this paper are listed in Appendix B.

A. Code-Tuple

We formalize a time-variant encoder with a finite number of code tables as a *code-tuple*. An m-code-tuple consists of m code tables $f_0, f_1 \dots, f_{m-1}$ from $\mathcal{S}$ to $\mathcal{C}^*$ and m mappings $\tau_0, \tau_1, \dots, \tau_{m-1}$ from $\mathcal{S}$ to $[m] := \{0, 1, 2, \dots, m-1\}$. The m mappings $\tau_0, \tau_1, \dots, \tau_{m-1}$ determine which code table to use to encode the i-th symbol x_i of a source sequence: if the previous symbol x_{i-1} is encoded with f_j, then the current symbol x_i is encoded with $f_{\tau_j(x_{i-1})}$.

Definition 1: Let m be a positive integer. An *m-code-tuple* $F(f_i, \tau_i : i \in [m])$ is a tuple of m mappings $f_i : \mathcal{S} \to \mathcal{C}^*, i \in [m]$ and m mappings $\tau_i : \mathcal{S} \to [m], i \in [m]$.

Let $\mathscr{F}^{(m)}$ denote the set of all m-code-tuples and define $\mathscr{F} := \mathscr{F}^{(1)} \cup \mathscr{F}^{(2)} \cup \mathscr{F}^{(3)} \cup \cdots$. An element of $\mathscr{F}$ is called a *code-tuple*. We sometimes write $F(f_i, \tau_i : i \in [m])$ as $F(f, \tau)$ or F for simplicity. For $F \in \mathscr{F}^{(m)}$, let $|F|$ denote the number of code tables of F, that is, $|F| := m$. We write $[|F|]$ as $[F]$ for simplicity.

TABLE III
THREE EXAMPLES OF A CODE-TUPLE $F^{(\alpha)}(f^{(\alpha)}, \tau^{(\alpha)})$, $F^{(\beta)}(f^{(\beta)}, \tau^{(\beta)})$, $F^{(\gamma)}(f^{(\gamma)}, \tau^{(\gamma)})$, AND $F^{(\delta)}(f^{(\delta)}, \tau^{(\delta)})$

$s \in \mathcal{S}$	$f_0^{(\alpha)}$	$\tau_0^{(\alpha)}$	$f_1^{(\alpha)}$	$\tau_1^{(\alpha)}$	$f_2^{(\alpha)}$	$\tau_2^{(\alpha)}$	$f_3^{(\alpha)}$	$\tau_3^{(\alpha)}$
a	λ	1	110	3	010	0	λ	3
b	000	1	λ	2	011	1	λ	3
c	001	2	110	1	10	2	λ	3

$s \in \mathcal{S}$	$f_0^{(\beta)}$	$\tau_0^{(\beta)}$	$f_1^{(\beta)}$	$\tau_1^{(\beta)}$	$f_2^{(\beta)}$	$\tau_2^{(\beta)}$
a	11	2	0110	1	10	2
b	λ	1	0110	2	11	1
c	101	1	01	1	1000	0
d	1011	1	0111	2	1001	1
e	1101	2	01110	2	11100	2

$s \in \mathcal{S}$	$f_0^{(\gamma)}$	$\tau_0^{(\gamma)}$	$f_1^{(\gamma)}$	$\tau_1^{(\gamma)}$	$f_2^{(\gamma)}$	$\tau_2^{(\gamma)}$
a	111	2	1100	1	01	2
b	0	1	1101	2	10	1
c	1010	1	10	1	000	0
d	10110	1	1111	2	0010	1
e	11011	2	11101	2	11001	2

$s \in \mathcal{S}$	$f_0^{(\delta)}$	$\tau_0^{(\delta)}$	$f_1^{(\delta)}$	$\tau_1^{(\delta)}$	$f_2^{(\delta)}$	$\tau_2^{(\delta)}$
a	111	2	1001	1	01	2
b	01	1	101	2	101	1
c	10101	1	01	1	000	0
d	101101	1	111	2	00101	1
e	11011	2	1101	2	11001	2

Example 1: Table III shows four examples of a code-tuple $F^{(\alpha)}(f^{(\alpha)}, \tau^{(\alpha)}) \in \mathscr{F}^{(4)}$ for $\mathcal{S} = \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$, and $F^{(\beta)}(f^{(\beta)}, \tau^{(\beta)}), F^{(\gamma)}(f^{(\gamma)}, \tau^{(\gamma)}), F^{(\delta)}(f^{(\delta)}, \tau^{(\delta)}) \in \mathscr{F}^{(3)}$ for $\mathcal{S} = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}, \mathrm{e}\}$.

In encoding $x_1x_2 \ldots x_n \in \mathcal{S}^*$ with $F(f, \tau) \in \mathscr{F}$, the mappings $\tau_0, \tau_1, \ldots, \tau_{|F|-1}$ determine which code table to use to encode x_i for $i = 2, 3, \ldots, n$. However, there are choices of which code table to use for the first symbol x_1. For $i \in [F]$ and $\boldsymbol{x} \in \mathcal{S}^*$, we define $f_i^*(\boldsymbol{x}) \in \mathcal{C}^*$ as the codeword sequence in the case where x_1 is encoded with f_i. Namely, $f_i^*(x_1x_2x_3 \ldots) = f_i(x_1)f_{\tau_i(x_1)}(x_2)f_{\tau_{\tau_i(x_1)}(x_2)}(x_3) \ldots$. Also, we define $\tau_i^*(\boldsymbol{x}) \in [F]$ as the index of the code table to be used next after encoding $\boldsymbol{x}$ in the case where x_1 is encoded with f_i. We give formal definitions of f_i^* and τ_i^* in the following Definition 2 as recursive formulas.

Definition 2: For $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$, we define a mapping $f_i^* : \mathcal{S}^* \to \mathcal{C}^*$ and a mapping $\tau_i^* : \mathcal{S}^* \to [F]$ as follows.

$$f_i^*(\boldsymbol{x}) = \begin{cases} \lambda & \text{if } \boldsymbol{x} = \lambda, \\ f_i(x_1)f^*_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \neq \lambda, \end{cases} \tag{5}$$

$$\tau_i^*(\boldsymbol{x}) = \begin{cases} i & \text{if } \boldsymbol{x} = \lambda, \\ \tau^*_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \neq \lambda, \end{cases} \tag{6}$$

for $\boldsymbol{x} = x_1x_2 \ldots x_n \in \mathcal{S}^*$.

Example 2: Let $F(f, \tau)$ be $F^{(\beta)}(f^{(\beta)}, \tau^{(\beta)})$ of Table III. We have

$$\begin{aligned} f_2^*(\mathrm{baed}) &= f_2(\mathrm{b})f_1^*(\mathrm{aed}) && (7)\\ &= f_2(\mathrm{b})f_1(\mathrm{a})f_1^*(\mathrm{ed}) && (8)\\ &= f_2(\mathrm{b})f_1(\mathrm{a})f_1(\mathrm{e})f_2^*(\mathrm{d}) && (9)\\ &= f_2(\mathrm{b})f_1(\mathrm{a})f_1(\mathrm{e})f_2(\mathrm{d})f_1^*(\lambda) && (10)\\ &= f_2(\mathrm{b})f_1(\mathrm{a})f_1(\mathrm{e})f_2(\mathrm{d}) && (11)\\ &= 110110011101001. && (12) \end{aligned}$$

$$\tau_2^*(\mathrm{baed}) = \tau_1^*(\mathrm{aed}) = \tau_1^*(\mathrm{ed}) = \tau_2^*(\mathrm{d}) = \tau_1^*(\lambda) = 1. \tag{13}$$

Directly from Definition 2, we obtain the following lemma.

Lemma 1: For any $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$, the following conditions (i)–(iii) hold.

(i) For any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^*$, $f_i^*(\boldsymbol{xy}) = f_i^*(\boldsymbol{x})f^*_{\tau_i^*(\boldsymbol{x})}(\boldsymbol{y})$.

(ii) For any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^*$, $\tau_i^*(\boldsymbol{xy}) = \tau^*_{\tau_i^*(\boldsymbol{x})}(\boldsymbol{y})$.

(iii) For any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^*$, if $\boldsymbol{x} \preceq \boldsymbol{y}$, then $f_i^*(\boldsymbol{x}) \preceq f_i^*(\boldsymbol{y})$.

For the code-tuple $F^{(\alpha)}$ in Table III, we can see that $f_3^{(\alpha)*}(\boldsymbol{x}) = \lambda$ for any $\boldsymbol{x} \in \mathcal{S}^*$. To distinguish such abnormal and useless code-tuples from the others, we introduce a class $\mathscr{F}_{\mathrm{ext}}$ in the following Definition 3.

Definition 3: We define $\mathscr{F}_{\mathrm{ext}}$ as the set of all $F(f, \tau) \in \mathscr{F}$ such that for any $i \in [F]$, there exists $\boldsymbol{x} \in \mathcal{S}^*$ such that $|f_i^*(\boldsymbol{x})| > 0$.

Directly from Definition 3, for any $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}$, $i \in [F]$ and an integer $k \geq 0$, there exists $\boldsymbol{x} \in \mathcal{S}^*$ such that $|f_i^*(\boldsymbol{x})| \geq k$. Namely, we can extend $f_i^*(\boldsymbol{x})$ as long as we want by appending symbols to $\boldsymbol{x}$ appropriately. The subscription "ext" of $\mathscr{F}_{\mathrm{ext}}$ is an abbreviation of "extendable."

Example 3: Consider $F^{(\alpha)}$, $F^{(\beta)}$, $F^{(\gamma)}$, and $F^{(\delta)}$ of Table III. We have $F^{(\alpha)} \notin \mathscr{F}_{\mathrm{ext}}$ because $|f_3^{(\alpha)*}(\boldsymbol{x})| = 0$ for any $\boldsymbol{x} \in \mathcal{S}^*$. On the other hand, $F^{(\beta)}, F^{(\gamma)}, F^{(\delta)} \in \mathscr{F}_{\mathrm{ext}}$.

B. k-Bit Delay Decodable Code-Tuple

Let $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$. Consider a situation that a source sequence $\boldsymbol{x}' \in \mathcal{S}^*$ is encoded with F starting from the code table f_i. Then the source sequence $\boldsymbol{x}'$ is encoded to the codeword sequence $f_i^*(\boldsymbol{x}')$, and the decoder reads it bit by bit from the beginning. Let $\boldsymbol{b} \preceq f(\boldsymbol{x}')$ be the sequence the decoder has read by a certain moment of the decoding process. If $\boldsymbol{b} = f(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{S}^*$, then there are two possible cases, $\boldsymbol{x} \preceq \boldsymbol{x}'$ and $\boldsymbol{x} \not\preceq \boldsymbol{x}'$. We say that F is *k-bit delay decodable* if it is always possible for the decoder to distinguish the two cases, $\boldsymbol{x} \preceq \boldsymbol{x}'$ and $\boldsymbol{x} \not\preceq \boldsymbol{x}'$, by reading the following k bits $\boldsymbol{c} \in \mathcal{C}^k$ of the codeword sequence $f(\boldsymbol{x})$, that is, for any pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^k$, the decoder can distinguish the two cases, $\boldsymbol{x} \preceq \boldsymbol{x}'$ and $\boldsymbol{x} \not\preceq \boldsymbol{x}'$ according to the pair $(\boldsymbol{x}, \boldsymbol{c})$. Thus, F is k-bit delay decodable if and only if for any pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^k$, it holds that $(\boldsymbol{x}, \boldsymbol{c})$ is f_i^**-positive* or f_i^**-negative* defined as follows.

Definition 4: Let $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$.

(i) A pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^*$ is said to be f_i^**-positive* if for any $\boldsymbol{x}' \in \mathcal{S}^*$, if $f_i^*(\boldsymbol{x})\boldsymbol{c} \preceq f_i^*(\boldsymbol{x}')$, then $\boldsymbol{x} \preceq \boldsymbol{x}'$.

(ii) A pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^*$ is said to be f_i^**-negative* if for any $\boldsymbol{x}' \in \mathcal{S}^*$, if $f_i^*(\boldsymbol{x})\boldsymbol{c} \preceq f_i^*(\boldsymbol{x}')$, then $\boldsymbol{x} \not\preceq \boldsymbol{x}'$.

Definition 5: Let $F(f, \tau) \in \mathscr{F}$ and let $k \geq 0$ be an integer. The code-tuple F is said to be *k-bit delay decodable* if for any $i \in [F]$ and $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^k$, the pair $(\boldsymbol{x}, \boldsymbol{c})$ is f_i^*-positive or f_i^*-negative. For an integer $k \geq 0$, we define $\mathscr{F}_{k\text{-dec}}$ as the set of all k-bit delay decodable code-tuples.

Note that it is possible that a pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^*$ is f_i^*-positive and f_i^*-negative simultaneously. A pair $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^*$ is f_i^*-positive and f_i^*-negative simultaneously if and only if there is no sequence $\boldsymbol{x}'$ satisfying $f_i^*(\boldsymbol{x})\boldsymbol{c} \preceq f_i^*(\boldsymbol{x}')$.

In fact, the classes $\mathscr{F}_{k\text{-dec}}, k = 0, 1, 2, \ldots$ form a hierarchical structure $\mathscr{F}_{0\text{-dec}} \subseteq \mathscr{F}_{1\text{-dec}} \subseteq \mathscr{F}_{2\text{-dec}} \subseteq \cdots$. Namely, the following Lemma 2 holds.

Lemma 2: For any two non-negative integers k, k' such that $k \le k'$, we have $\mathscr{F}_{k\text{-dec}} \subseteq \mathscr{F}_{k'\text{-dec}}$.

Proof of Lemma 2: Let $F(f, \tau) \in \mathscr{F}_{k\text{-dec}}$. Fix $i \in [F]$ and $(\boldsymbol{x}, \boldsymbol{c}') \in \mathcal{S}^* \times \mathcal{C}^{k'}$ arbitrarily. It suffices to prove that $(\boldsymbol{x}, \boldsymbol{c}')$ is f_i^*-positive or f_i^*-negative.

Let $\boldsymbol{c}$ be the prefix of $\boldsymbol{c}'$ of length k. Then for any $\boldsymbol{x}' \in \mathcal{S}^*$ such that $f_i^*(\boldsymbol{x})\boldsymbol{c}' \preceq f_i^*(\boldsymbol{x}')$, we have $f_i^*(\boldsymbol{x})\boldsymbol{c} \preceq f_i^*(\boldsymbol{x})\boldsymbol{c}' \preceq f_i^*(\boldsymbol{x}')$. Namely, $f_i^*(\boldsymbol{x})\boldsymbol{c}' \preceq f_i^*(\boldsymbol{x}')$ implies $f_i^*(\boldsymbol{x})\boldsymbol{c} \preceq f_i^*(\boldsymbol{x}')$. Hence, from Definition 4, if $(\boldsymbol{x}, \boldsymbol{c}')$ is f_i^*-positive (resp. f_i^*-negative), then also $(\boldsymbol{x}, \boldsymbol{c})$ is f_i^*-positive (resp. f_i^*-negative), respectively. Therefore, it follows that $F(f, \tau) \in \mathscr{F}_{k'\text{-dec}}$ from $F(f, \tau) \in \mathscr{F}_{k\text{-dec}}$. ■

Example 4: Consider $F^{(\alpha)}, F^{(\beta)}, F^{(\gamma)}$, and $F^{(\delta)}$ of Table III. We have $F^{(\alpha)} \in \mathscr{F}_{2\text{-dec}} \setminus \mathscr{F}_{1\text{-dec}}$, $F^{(\beta)} \in \mathscr{F}_{1\text{-dec}} \setminus \mathscr{F}_{0\text{-dec}}$, and $F^{(\gamma)}, F^{(\delta)} \in \mathscr{F}_{0\text{-dec}}$.

We remark that a k-bit delay decodable code-tuple is not necessarily uniquely decodable. For example, for the code-tuple $F^{(\beta)} \in \mathscr{F}_{1\text{-dec}}$, we have $f_0^{(\beta)^*}(\text{ac}) = 111000 = f_0^{(\beta)^*}(\text{acb})$. In general, it is possible that the decoder cannot uniquely recover the last few symbols of the original source sequence because the length of the rest of the codeword sequence is less than k bits. In such a case, we should append additional information to uniquely decode the suffix in practical use.

However, as we show in the following Lemma 3, a 0-bit delay decodable code-tuple (i.e., an instantaneous code) is always uniquely decodable.

Lemma 3: For any $F(f, \tau) \in \mathscr{F}_{0\text{-dec}}$ and $i \in [F]$, the following conditions (i) and (ii) hold.

(i) For any $\boldsymbol{x} \in \mathcal{S}^*$, the pair $(\boldsymbol{x}, \lambda)$ is f_i^*-positive.
(ii) f_i^* is injective.

Proof of Lemma 3: (Proof of (i)) From $F \in \mathscr{F}_{0\text{-dec}}$, the pair $(\boldsymbol{x}, \lambda)$ is f_i^*-positive or f_i^*-negative. However, since $f_i^*(\boldsymbol{x}) \preceq f_i^*(\boldsymbol{x})$ and $\boldsymbol{x} \preceq \boldsymbol{x}$, the pair $(\boldsymbol{x}, \lambda)$ must be f_i^*-positive.

(Proof of (ii)) From (i), we have

$$\forall \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*, \left(f_i^*(\boldsymbol{x}) \preceq f_i^*(\boldsymbol{x}') \Rightarrow \boldsymbol{x} \preceq \boldsymbol{x}'\right). \tag{14}$$

Choose $\boldsymbol{y}, \boldsymbol{y}' \in \mathcal{S}^*$ such $f_i^*(\boldsymbol{y}) = f_i^*(\boldsymbol{y}')$ arbitrarily. Then we have $f_i^*(\boldsymbol{y}) \preceq f_i^*(\boldsymbol{y}')$ and $f_i^*(\boldsymbol{y}') \preceq f_i^*(\boldsymbol{y})$. From (14), we obtain $\boldsymbol{y} \preceq \boldsymbol{y}'$ and $\boldsymbol{y}' \preceq \boldsymbol{y}$, that is, $\boldsymbol{y} = \boldsymbol{y}'$. Consequently, f_i^* is injective. ■

For $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$, f_i is said to be *prefix-free* if for any $s, s' \in \mathcal{S}$, if $f_i(s) \preceq f_i(s')$, then $s = s'$. A 0-bit delay decodable code-tuple is characterized as a code-tuple all of which code tables are prefix-free.

Lemma 4: A code-tuple $F(f, \tau) \in \mathscr{F}$ satisfies $F \in \mathscr{F}_{0\text{-dec}}$ if and only if for any $i \in [F]$, f_i is prefix-free.

Proof of Lemma 4: (Proof of "only if") Assume $F \in \mathscr{F}_{0\text{-dec}}$ and choose $i \in [F]$ arbitrarily. From Lemma 3 (i), the pair $(\boldsymbol{x}, \lambda)$ is f_i^*-positive. Thus, (14) holds. In particular, we have

$$\forall s, s' \in \mathcal{S}, \left(f_i^*(s) \preceq f_i^*(s') \Rightarrow s \preceq s'\right). \tag{15}$$

Since $s \preceq s'$ implies $s = s'$, f_i is prefix-free.

(Proof of "if") Assume that for any $i \in [F]$, f_i is prefix-free. Fix $i \in [F]$ arbitrarily. To prove $F \in \mathscr{F}_{0\text{-dec}}$, it suffices to prove (14). We prove it by induction for $|\boldsymbol{x}|$.

For the case $|\boldsymbol{x}| = 0$, clearly we have $\boldsymbol{x} \preceq \boldsymbol{x}'$ for any $\boldsymbol{x}' \in \mathcal{S}^*$.

Let $l \ge 1$ and assume that (14) is true for the case $|\boldsymbol{x}| < l$ as the induction hypothesis. We prove (14) for the case $|\boldsymbol{x}| = l$. Choose $\boldsymbol{x}' \in \mathcal{S}^*$ such that $f_i^*(\boldsymbol{x}) \preceq f_i^*(\boldsymbol{x}')$ arbitrarily. Then by (5), we have

$$f_i(x_1) f^*_{\tau_i^*(x_1)}(\mathrm{suf}(\boldsymbol{x})) \preceq f_i(x'_1) f^*_{\tau_i^*(x'_1)}\left(\mathrm{suf}(\boldsymbol{x}')\right), \tag{16}$$

where $\boldsymbol{x} = x_1 x_2 \ldots x_n$, $\boldsymbol{x}' = x'_1 x'_2 \ldots x'_{n'}$. Thus, $f_i(x_1) \preceq f_i(x'_1)$ or $f_i(x_1) \succeq f_i(x'_1)$ holds. Hence, since f_i is prefix-free, we obtain

$$x_1 = x'_1. \tag{17}$$

From (16) and (17), we have $f_i(x_1) f^*_{\tau_i^*(x_1)}(\mathrm{suf}(\boldsymbol{x})) \preceq f_i(x_1) f^*_{\tau_i^*(x'_1)}(\mathrm{suf}(\boldsymbol{x}'))$. Thus, we have $f^*_{\tau_i^*(x_1)}(\mathrm{suf}(\boldsymbol{x})) \preceq f^*_{\tau_i^*(x'_1)}(\mathrm{suf}(\boldsymbol{x}'))$. From the induction hypothesis,

$$\mathrm{suf}(\boldsymbol{x}) \preceq \mathrm{suf}\left(\boldsymbol{x}'\right). \tag{18}$$

By (17) and (18), we obtain $\boldsymbol{x} \preceq \boldsymbol{x}'$. ■

C. Average Codeword Length of Code-Tuple

We introduce the average codeword length $L(F)$ of code-tuple F. Hereinafter, we fix a probability distribution μ of source symbols, that is, a real-valued function $\mu : \mathcal{S} \to \mathbb{R}$ satisfying $\sum_{s \in \mathcal{S}} \mu(s) = 1$ and $0 < \mu(s) \le 1$ for any $s \in \mathcal{S}$. Note that we exclude the case where $\mu(s) = 0$ for some $s \in \mathcal{S}$ from our consideration without loss of generality.

First, we define the transition probability $Q_{i,j}(F)$ for $F(f, \tau) \in \mathscr{F}$ and $i, j \in [F]$ as the probability of using f_j immediately after using f_i.

Definition 6: For $F(f, \tau) \in \mathscr{F}$ and $i, j \in [F]$, we define the *transition probability* $Q_{i,j}(F)$ as $Q_{i,j}(F) \coloneqq \sum_{s \in \mathcal{S}, \tau_i(s) = j} \mu(s)$. We also define the *transition probability matrix* $Q(F)$ as the following $|F| \times |F|$ matrix.

$$Q(F) \coloneqq \begin{bmatrix} Q_{0,0}(F) & Q_{0,1}(F) & \cdots & Q_{0,|F|-1}(F) \\ Q_{1,0}(F) & Q_{1,1}(F) & \cdots & Q_{1,|F|-1}(F) \\ \vdots & \vdots & \ddots & \vdots \\ Q_{|F|-1,0}(F) & Q_{|F|-1,1}(F) & \cdots & Q_{|F|-1,|F|-1}(F) \end{bmatrix}. \tag{19}$$

Fix $F \in \mathscr{F}$ and let $I_i \in [F]$ be the index of the code table used to encode the i-th symbol of the source sequence in encoding with F for $i = 1, 2, 3, \ldots$. Then $I_i \in [F], i = 1, 2, 3, \ldots$ is a Markov process with the transition probability matrix $Q(F)$. We consider a stationary distribution of the Markov process (i.e., the solution of the simultaneous equations (20) and (21)). The average codeword length $L(F)$ of a code-tuple F depends on a stationary distribution of the Markov process with $Q(F)$. Hence, to define $L(F)$ uniquely, a stationary distribution of the Markov process with $Q(F)$ must be unique. Thus, we define the class $\mathscr{F}_{\mathrm{reg}}$ of all code-tuples such that the Markov process has a unique stationary distribution.

Definition 7: A code-tuple $F \in \mathscr{F}$ is said to be *regular* if the following simultaneous equations (20) and (21) have a unique solution $\boldsymbol{\pi} = (\pi_0, \pi_1, \ldots, \pi_{|F|-1})$.

$$\begin{cases} \boldsymbol{\pi} Q(F) = \boldsymbol{\pi}, & (20) \\ \displaystyle\sum_{i \in [F]} \pi_i = 1. & (21) \end{cases}$$

We define $\mathscr{F}_{\mathrm{reg}}$ as the set of all regular code-tuples. For $F \in \mathscr{F}_{\mathrm{reg}}$, we define $\boldsymbol{\pi}(F) = (\pi_0(F), \pi_1(F), \ldots, \pi_{|F|-1}(F))$ as the unique solution of the simultaneous equations (20) and (21).

For any $F \in \mathscr{F}_{\mathrm{reg}}$, the asymptotical performance (average codeword length per symbol) does not depend on which code table we start encoding: the average codeword length $L(F)$ of a regular code tuple $F \in \mathscr{F}_{\mathrm{reg}}$ is the weighted sum of average codeword lengths of the code tables $f_0, f_1, \ldots, f_{|F|-1}$ weighted by the stationary distribution $\boldsymbol{\pi}(F)$.

Definition 8: For $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$, we define the *average codeword length $L_i(F)$ of a single code table $f_i : \mathcal{S} \to \mathcal{C}^*$* as $L_i(F) := \sum_{s \in \mathcal{S}} |f_i(s)| \cdot \mu(s)$. For $F \in \mathscr{F}_{\mathrm{reg}}$, we define the *average codeword length $L(F)$ of code-tuple F* as $L(F) := \sum_{i \in [F]} \pi_i(F) L_i(F)$.

Example 5: Let $F(f, \tau)$ be $F^{(\beta)}(f^{(\beta)}, \tau^{(\beta)})$ of Table III and let $(\mu(\mathrm{a}), \mu(\mathrm{b}), \mu(\mathrm{c}), \mu(\mathrm{d}), \mu(\mathrm{e})) = (0.1, 0.2, 0.2, 0.2, 0.3)$. We have

$$Q(F) = \begin{bmatrix} 0 & 0.6 & 0.4 \\ 0 & 0.3 & 0.7 \\ 0.2 & 0.4 & 0.4 \end{bmatrix}, \tag{22}$$

$$L_0(F) = 2.8, \quad L_1(F) = 3.9, \quad L_2(F) = 3.7. \tag{23}$$

Also, we obtain $\boldsymbol{\pi}(F) = (7/68, 26/68, 35/68)$ by solving the simultaneous equations (20) and (21). Therefore, the average codeword length $L(F)$ of the code-tuple F is given as

$$L(F) = \pi_0(F)L_0(F) + \pi_1(F)L_1(F) + \pi_2(F)L_2(F) \tag{24}$$

$$= \frac{7 \cdot 2.8 + 26 \cdot 3.9 + 35 \cdot 3.7}{68} \tag{25}$$

$$\approx 3.683823. \tag{26}$$

III. The Optimality of Huffman Code

In this section, we prove the following Theorem 1 as the main result of this paper.

Theorem 1: For any $F(f, \tau) \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, we have $L(F) \geq L_{\mathrm{Huff}}$, where L_{Huff} is the average codeword length of the Huffman code.

The outline of the proof is as follows. First, we define an operation called *rotation* which transforms a code-tuple $F \in \mathscr{F}_{\mathrm{ext}}$ into another code-tuple $\widehat{F} \in \mathscr{F}$. Then we show that any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$ can be transformed into some $F' \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$ by rotation in a repetitive manner without changing the average codeword length. Hence, without loss of generality, we can assume that a given code-tuple is in $\mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$, in particular, 0-bit delay decodable. Then we complete the proof of Theorem 1 using McMillan's theorem in the context of 0-bit delay decodable code. This section consists of the following four subsections:

1) In Section III-A, we introduce *rotation* which transforms a code-tuple $F \in \mathscr{F}_{\mathrm{ext}}$ into another code-tuple $\widehat{F} \in \mathscr{F}$.
2) In Section III-B, we show that for any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, we have $\widehat{F} \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$ and $L(\widehat{F}) = L(F)$. Namely, the rotation preserves "the key properties" of a code-tuple.
3) In Section III-C, we prove that for any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, there exists $F' \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$ such that $L(F') = L(F)$. Namely, any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$ can be replaced with some $F' \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$.
4) In Section III-D, we complete the proof of Theorem 1 using McMillan's theorem.

TABLE IV
$\mathcal{P}_{F,i}, i \in [F]$ AND $d_{F,i}, i \in [F]$ FOR THE CODE-TUPLES $F^{(\alpha)}, F^{(\beta)}, F^{(\gamma)}$, AND $F^{(\delta)}$ IN TABLE III

F	$\mathcal{P}_{F,0}$	$d_{F,0}$	$\mathcal{P}_{F,1}$	$d_{F,1}$	$\mathcal{P}_{F,2}$	$d_{F,2}$	$\mathcal{P}_{F,3}$	$d_{F,3}$
$F^{(\alpha)}$	$\{0,1\}$	-	$\{0,1\}$	-	$\{0,1\}$	-	$\emptyset$	-
$F^{(\beta)}$	$\{0,1\}$	λ	$\{0\}$	0	$\{1\}$	1	-	-
$F^{(\gamma)}$	$\{0,1\}$	λ	$\{1\}$	1	$\{0,1\}$	λ	-	-
$F^{(\delta)}$	$\{0,1\}$	λ	$\{0,1\}$	λ	$\{0,1\}$	λ	-	-

A. Rotation

As stated above, the first step in the proof is to define rotation. To describe the definition of rotation, we introduce the following Definition 9.

Definition 9: For $F(f, \tau) \in \mathscr{F}$ and $i \in [F]$, we define $\mathcal{P}_{F,i}$ as $\mathcal{P}_{F,i} := \{c \in \mathcal{C} : {}^{\exists}\boldsymbol{x} \in \mathcal{S}^*, f_i^*(\boldsymbol{x}) \succeq c\}$, that is, $\mathcal{P}_{F,i}$ is the set of all $c \in \mathcal{C}$ which is the first bit of $f_i^*(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{S}^*$.

For $F \in \mathscr{F}_{\mathrm{ext}}$ and $i \in [F]$, we define $d_{F,i}$ as follows.

$$d_{F,i} = \begin{cases} 0 & \text{if } \mathcal{P}_{F,i} = \{0\}, \\ 1 & \text{if } \mathcal{P}_{F,i} = \{1\}, \\ \lambda & \text{if } \mathcal{P}_{F,i} = \{0, 1\}. \end{cases} \tag{27}$$

Note that for any $F \in \mathscr{F}_{\mathrm{ext}}$ and $i \in [F]$, we have $\mathcal{P}_{F,i} \neq \emptyset$.

Example 6: Table IV shows $\mathcal{P}_{F,i}, i \in [F]$ and $d_{F,i}, i \in [F]$ for the code-tuples $F^{(\alpha)}, F^{(\beta)}, F^{(\gamma)}$, and $F^{(\delta)}$ in Table III. Note that $d_{F^{(\alpha)},0}, d_{F^{(\alpha)},1}$ and $d_{F^{(\alpha)},2}$ are not defined since $F^{(\alpha)} \notin \mathscr{F}_{\mathrm{ext}}$.

Then the rotation is defined as follows.

Definition 10: For $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}$, we define $\widehat{F}(\widehat{f}, \widehat{\tau}) \in \mathscr{F}$ as follows.

For $i \in [F]$ and $s \in \mathcal{S}$,

$$\widehat{f}_i(s) = \begin{cases} f_i(s) d_{F,\tau_i(s)} & \text{if } \mathcal{P}_{F,i} = \{0, 1\}, \\ \mathrm{suf}(f_i(s) d_{F,\tau_i(s)}) & \text{if } \mathcal{P}_{F,i} \neq \{0, 1\}, \end{cases} \tag{28}$$

$$\widehat{\tau}_i(s) = \tau_i(s). \tag{29}$$

The operation which transforms a given $F \in \mathscr{F}_{\mathrm{ext}}$ into $\widehat{F} \in \mathscr{F}$ defined above is called *rotation*.

Example 7: In Table III, $F^{(\gamma)}$ is obtained by applying rotation to $F^{(\beta)}$, that is, $F^{(\gamma)} = \widehat{F^{(\beta)}}$. Also, $F^{(\delta)}$ is obtained by applying rotation to $F^{(\gamma)}$, that is, $F^{(\delta)} = \widehat{F^{(\gamma)}}$. Furthermore, we obtain $F^{(\delta)}$ itself applying rotation to $F^{(\delta)}$, that is, $F^{(\delta)} = \widehat{F^{(\delta)}}$.

Directly from Definition 10, we can see that for any $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}, i \in [F]$, and $s \in \mathcal{S}$, we have

$$d_{F,i} \widehat{f}_i(s) = f_i(s) d_{F,\tau_i(s)}. \tag{30}$$

We show that (30) is generalized to the following Lemma 5.

Lemma 5: For any $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}, i \in [F]$, and $\boldsymbol{x} \in \mathcal{S}^*$,

$$d_{F,i} \widehat{f}_i^*(\boldsymbol{x}) = f_i^*(\boldsymbol{x}) d_{F,\tau_i^*(\boldsymbol{x})}. \tag{31}$$

Proof of Lemma 5: We prove the lemma by induction for $|\boldsymbol{x}|$.

For the case $|\boldsymbol{x}| = 0$, we have $d_{F,i} \widehat{f}_i^*(\boldsymbol{x}) = d_{F,i} \widehat{f}_i^*(\lambda) = d_{F,i} \lambda = \lambda d_{F,i} = f_i^*(\lambda) d_{F,\tau_i^*(\lambda)} = f_i^*(\boldsymbol{x}) d_{F,\tau_i^*(\boldsymbol{x})}$, where the second and fourth equalities are from (5). Hence, (31) holds.

Let $l \geq 1$ and assume that (31) holds for any $\boldsymbol{x}' \in \mathcal{S}^*$ such that $|\boldsymbol{x}'| < l$ as the induction hypothesis. We prove that (31) holds for $\boldsymbol{x} := x_1 x_2 \ldots x_l \in \mathcal{S}^*$. We have

$$\begin{aligned} d_{F,i}\widehat{f_i^*}(\boldsymbol{x}) &= d_{F,i}\widehat{f_i}(x_1)\widehat{f^*}_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x})) && (32)\\ &= f_i(x_1)d_{F,\tau_i(x_1)}\widehat{f^*}_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x})) && (33)\\ &= f_i(x_1)f^*_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x}))d_{F,\tau^*_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x}))} && (34)\\ &= f_i(x_1)f^*_{\tau_i(x_1)}(\mathrm{suf}(\boldsymbol{x}))d_{F,\tau_i^*(\boldsymbol{x})} && (35)\\ &= f_i^*(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})}, && (36)\end{aligned}$$

where the first equality is from (5), the second equality is from (30), the third equality is from the induction hypothesis, the fourth equality is from (6) and the fifth equality is from (5). Hence, (31) holds for $\boldsymbol{x} := x_1 x_2 \ldots x_l \in \mathcal{S}^*$. ■

Example 8: Let $F(f, \tau)$ be $F^{(\beta)}$ of Table III. As seen in Example 7, we have $\widehat{F}(\widehat{f}, \widehat{\tau}) = F^{(\gamma)}$. We can see $d_{F,2}\widehat{f_2^*}(\text{baed})$ $=$ $d_{F,2}\widehat{f_2}(\text{b})\widehat{f_1}(\text{a})\widehat{f_1}(\text{e})\widehat{f_2}(\text{d})$ $=$ 1101100111010010, and $f_2^*(\text{baed})d_{F,\tau_2^*(\text{baed})} = f_2^*(\text{baed})d_{F,1} = 1101100111010010$. Hence, we can confirm $d_{F,2}\widehat{f_2^*}(\text{baed}) = f_2^*(\text{baed})d_{F,\tau_2^*(\text{baed})}$.

B. Rotation Preserves the Key Properties

In the previous subsection, we introduced rotation, which transforms a given $F \in \mathscr{F}_{\mathrm{ext}}$ into the $\widehat{F} \in \mathscr{F}$. In this subsection, we prove that if $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, then $\widehat{F} \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$ and $L(\widehat{F}) = L(F)$. To prove it, we show the following Lemmas 6–8.

Lemma 6: For any integer $k \geq 0$ and $F(f, \tau) \in \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{ext}}$, we have $\widehat{F}(\widehat{f}, \widehat{\tau}) \in \mathscr{F}_{k\text{-dec}}$.

Lemma 7: For any $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}$, we have $\widehat{F}(\widehat{f}, \widehat{\tau}) \in \mathscr{F}_{\mathrm{ext}}$.

Lemma 8: For any $F \in \mathscr{F}_{\mathrm{reg}}$, we have $\widehat{F} \in \mathscr{F}_{\mathrm{reg}}$ and $L(\widehat{F}) = L(F)$.

The proof of Lemma 6 relies on Lemma 9 whose proof is relegated to Appendix A.

Lemma 9: Let $F(f, \tau) \in \mathscr{F}_{\mathrm{ext}}$. There exists no tuple $(k, i, \boldsymbol{x}, \boldsymbol{x}')$ satisfying all of the following conditions (i)–(iii), where k is a non-negative integer, $i \in [F]$, and $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$: (i) $F \in \mathscr{F}_{k\text{-dec}}$, (ii) $|\widehat{f_i^*}(\boldsymbol{x})| + k \leq |\widehat{f_i^*}(\boldsymbol{x}')|$, and (iii) $\boldsymbol{x}' \preceq \boldsymbol{x}$ and $\boldsymbol{x} \neq \boldsymbol{x}'$.

Proof of Lemma 6: Fix $i \in [F]$ and $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{S}^* \times \mathcal{C}^k$ arbitrarily. Also, choose $\boldsymbol{x}' \in \mathcal{S}^*$ such that $\widehat{f_i^*}(\boldsymbol{x})\boldsymbol{c} \preceq \widehat{f_i^*}(\boldsymbol{x}')$ arbitrarily. Then, we have $d_{F,i}\widehat{f_i^*}(\boldsymbol{x})\boldsymbol{c} \preceq d_{F,i}\widehat{f_i^*}(\boldsymbol{x}')$. From Lemma 5, we have $f_i^*(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})}\boldsymbol{c} \preceq f_i^*(\boldsymbol{x}')d_{F,\tau_i^*(\boldsymbol{x}')}$. From (27), there exists $\boldsymbol{x}'' \in \mathcal{S}^*$ such that $d_{F,\tau_i^*(\boldsymbol{x}')} \preceq f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{x}'')$. For such $\boldsymbol{x}''$, we have

$$\begin{aligned} f_i^*(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})}\boldsymbol{c} &\preceq f_i^*(\boldsymbol{x}')d_{F,\tau_i^*(\boldsymbol{x}')} && (37)\\ &\preceq f_i^*(\boldsymbol{x}')f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{x}'') && (38)\\ &= f_i^*(\boldsymbol{x}'\boldsymbol{x}''), && (39)\end{aligned}$$

where the last equality is from Lemma 1 (i). Therefore, we obtain

$$f_i^*(\boldsymbol{x})\boldsymbol{c}' \preceq f_i^*(\boldsymbol{x}'\boldsymbol{x}''), \quad (40)$$

where $\boldsymbol{c}'$ is the prefix of length of k of $d_{F,\tau_i^*(\boldsymbol{x})}\boldsymbol{c}$.

In general, exactly one of the following conditions (a)–(c) holds: (a) $\boldsymbol{x} \preceq \boldsymbol{x}'$, (b) $\boldsymbol{x} \neq \boldsymbol{x}'$ and $\boldsymbol{x} \succeq \boldsymbol{x}'$, and (c) $\boldsymbol{x} \not\preceq \boldsymbol{x}'$ and $\boldsymbol{x} \not\succeq \boldsymbol{x}'$. However, now (b) is impossible from Lemma 9. Thus, it suffices to consider the case where either (a) or (c) holds.

From $F \in \mathscr{F}_{k\text{-dec}}$, $(\boldsymbol{x}, \boldsymbol{c}')$ is f_i^*-positive or f_i^*-negative. If $(\boldsymbol{x}, \boldsymbol{c}')$ is f_i^*-positive (resp. f_i^*-negative), then we have $\boldsymbol{x} \preceq \boldsymbol{x}'\boldsymbol{x}''$ (resp. $\boldsymbol{x} \not\preceq \boldsymbol{x}'\boldsymbol{x}''$) from (40). This implies that (a) $\boldsymbol{x} \preceq \boldsymbol{x}'$ (resp. (c) $\boldsymbol{x} \not\preceq \boldsymbol{x}'$ and $\boldsymbol{x} \not\succeq \boldsymbol{x}'$) holds since (b) is impossible. Since $\boldsymbol{x}'$ is chosen arbitrarily, the pair $(\boldsymbol{x}, \boldsymbol{c})$ is $\widehat{f_i^*}$-positive (resp. $\widehat{f_i^*}$-negative). Therefore, we have $\widehat{F} \in \mathscr{F}_{k\text{-dec}}$. ■

Proof of Lemma 7: Fix $i \in [F]$ arbitrarily. From $F \in \mathscr{F}_{\mathrm{ext}}$, there exists $\boldsymbol{x} \in \mathcal{S}^*$ such that $|f_i^*(\boldsymbol{x})| \geq 2$. For such $\boldsymbol{x}$, from Lemma 5, we have $d_{F,i}\widehat{f_i^*}(\boldsymbol{x}) = f_i^*(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})}$. Hence, $|\widehat{f_i^*}(\boldsymbol{x})| = |f_i^*(\boldsymbol{x})| + |d_{F,\tau_i^*(\boldsymbol{x})}| - |d_{F,i}|$. From $|f_i^*(\boldsymbol{x})| \geq 2$, $|d_{F,\tau_i^*(\boldsymbol{x})}| \geq 0$, and $|d_{F,i}| \leq 1$, we obtain $|\widehat{f_i^*}(\boldsymbol{x})| \geq 1$. Therefore, we have $\widehat{F} \in \mathscr{F}_{\mathrm{ext}}$. ■

Proof of Lemma 8: From (29), for any $i, j \in [F]$, we have $Q_{i,j}(\widehat{F}) = Q_{i,j}(F)$. Thus, we have $\widehat{F} \in \mathscr{F}_{\mathrm{reg}}$, and for any $i \in [F]$, we have

$$\pi_i(\widehat{F}) = \pi_i(F). \quad (41)$$

Also, from (27) and (28), for any $i \in [F]$, we obtain

$$\begin{aligned} L_i(\widehat{F}) &= \begin{cases} \sum_{s\in\mathcal{S}} |f_i(s)d_{F,\tau_i(s)}| \cdot \mu(s) & \text{if } \mathcal{P}_{F,i} = \{0,1\},\\ \sum_{s\in\mathcal{S}} |\mathrm{suf}(f_i(s)d_{F,\tau_i(s)})| \cdot \mu(s) & \text{if } \mathcal{P}_{F,i} \neq \{0,1\}, \end{cases} && (42)\\ &= \begin{cases} L_i(F) + \sum_{s\in\mathcal{S}} d_{F,\tau_i(s)} \cdot \mu(s) & \text{if } \mathcal{P}_{F,i} = \{0,1\},\\ L_i(F) + \sum_{s\in\mathcal{S}} d_{F,\tau_i(s)} \cdot \mu(s) - 1 & \text{if } \mathcal{P}_{F,i} \neq \{0,1\}, \end{cases} && (43)\\ &= \begin{cases} L_i(F) + \sum_{j\in\mathcal{B}} Q_{i,j}(F) & \text{if } i \notin \mathcal{B},\\ L_i(F) + \sum_{j\in\mathcal{B}} Q_{i,j}(F) - 1 & \text{if } i \in \mathcal{B}, \end{cases} && (44)\end{aligned}$$

where $\mathcal{B} := \{i \in [F] : \mathcal{P}_{F,i} \neq \{0,1\}\}$.

Therefore, we have

$$\begin{aligned} L(\widehat{F}) &= \sum_{i\in[F]} \pi_i(\widehat{F})L_i(\widehat{F}) && (45)\\ &= \sum_{i\in[F]\setminus\mathcal{B}} \pi_i(\widehat{F})L_i(\widehat{F}) + \sum_{i\in\mathcal{B}} \pi_i(\widehat{F})L_i(\widehat{F}) && (46)\\ &= \sum_{i\in[F]\setminus\mathcal{B}} \pi_i(F)\left(L_i(F) + \sum_{j\in\mathcal{B}} Q_{i,j}(F)\right)\\ &\quad + \sum_{i\in\mathcal{B}} \pi_i(F)\left(L_i(F) + \sum_{j\in\mathcal{B}} Q_{i,j}(F) - 1\right) && (47)\\ &= \sum_{i\in[F]\setminus\mathcal{B}} \pi_i(F)L_i(F) + \sum_{i\in\mathcal{B}} \pi_i(F)L_i(F)\\ &\quad + \sum_{i\in[F]\setminus\mathcal{B}}\sum_{j\in\mathcal{B}} \pi_i(F)Q_{i,j}(F) + \sum_{i\in\mathcal{B}}\sum_{j\in\mathcal{B}} \pi_i(F)Q_{i,j}(F)\\ &\quad - \sum_{i\in\mathcal{B}} \pi_i(F) && (48)\\ &= \sum_{i\in[F]} \pi_i(F)L_i(F) + \sum_{i\in[F]}\sum_{j\in\mathcal{B}} \pi_i(F)Q_{i,j}(F) - \sum_{j\in\mathcal{B}} \pi_j(F) && (49)\\ &= \sum_{i\in[F]} \pi_i(F)L_i(F) + \sum_{i\in[F]}\sum_{j\in\mathcal{B}} \pi_i(F)Q_{i,j}(F)\\ &\quad - \sum_{j\in\mathcal{B}}\sum_{i\in[F]} \pi_i(F)Q_{i,j}(F) && (50)\end{aligned}$$

TABLE V
$L(F^{(\beta)}), L(F^{(\gamma)})$, AND $L(F^{(\delta)})$ ARE EQUAL

F	$L_0(F)$	$L_1(F)$	$L_2(F)$	$\pi_0(F)$	$\pi_1(F)$	$\pi_2(F)$	$L(F)$
$F^{(\beta)}$	2.8	3.9	3.7	7/68	26/68	35/68	3.683823...
$F^{(\gamma)}$	3.8	3.9	3.5	7/68	26/68	35/68	3.683823...
$F^{(\delta)}$	4.4	3.2	3.9	7/68	26/68	35/68	3.683823...

$$= \sum_{i\in[F]} \pi_i(F)L_i(F) \tag{51}$$
$$= L(F), \tag{52}$$

where the third equality is from (41) and (44), and the sixth equality is from (20). ∎

Example 9: Let $\mathcal{S} = \{\mathrm{a, b, c, d, e}\}$ and $(\mu(\mathrm{a}), \mu(\mathrm{b}), \mu(\mathrm{c}), \mu(\mathrm{d}), \mu(\mathrm{e})) = (0.1, 0.2, 0.2, 0.2, 0.3)$. Table V shows that $L(F^{(\beta)}), L(F^{(\gamma)})$, and $L(F^{(\delta)})$ are equal. We can see that the operation of rotation does not change the average codeword length for this example.

Summing up Lemmas 6–8, we obtain the next Lemma 10.

Lemma 10: For any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, we have $\widehat{F} \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$ and $L(\widehat{F}) = L(F)$.

C. Any $F \in \mathscr{F}_{reg} \cap \mathscr{F}_{ext} \cap \mathscr{F}_{1\text{-}dec}$ Can Be Replaced With Some $F' \in \mathscr{F}_{reg} \cap \mathscr{F}_{ext} \cap \mathscr{F}_{0\text{-}dec}$

The goal of this subsection is to prove Lemma 13 that for any $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}}$, there exists $F' \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$ such that $L(F') = L(F)$. To prove it, we show Lemma 11 that for any integer $k \geq 0$ and $F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}}$, there exists $F' \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{fork}}$ such that $L(F') = L(F)$, where $\mathscr{F}_{\mathrm{fork}}$ is defined in the following Definition 11. Then we prove the desired Lemma 13 arguing $\mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{1\text{-dec}} \cap \mathscr{F}_{\mathrm{fork}} \subseteq \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{0\text{-dec}}$.

Definition 11: We define $\mathscr{F}_{\mathrm{fork}}$ as

$$\mathscr{F}_{\mathrm{fork}} := \left\{F \in \mathscr{F} : {}^{\forall} i \in [F], \mathcal{P}_{F,i} = \{0, 1\}\right\}, \tag{53}$$

that is, $\mathscr{F}_{\mathrm{fork}}$ is the set of all code-tuples F such that $\mathcal{P}_{F,0} = \mathcal{P}_{F,1} = \cdots = \mathcal{P}_{F,|F|-1} = \{0, 1\}$.

Example 10: Consider the code-tuples of Table III. From Example 6, we have $F^{(\alpha)}, F^{(\beta)}, F^{(\gamma)} \notin \mathscr{F}_{\mathrm{fork}}$ and $F^{(\delta)} \in \mathscr{F}_{\mathrm{fork}}$.

The following Lemma 11 guarantees that we can assume that a given code-tuple is in $\mathscr{F}_{\mathrm{fork}}$ without loss of generality.

Lemma 11: For any integer $k \geq 0$ and $F(f, \tau) \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}}$, there exists $F'(f', \tau') \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{fork}}$ such that $L(F') = L(F)$.

To prove Lemma 11, for an integer $k \geq 0$, $F \in \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{ext}}$, and $i \in [F]$, we define $l_{F,i}$ as

$$l_{F,i} := \min\left\{|f_i^*(\boldsymbol{x}) \wedge f_i^*(\boldsymbol{x}')| : \boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*, f_i^*(\boldsymbol{x}) \not\precsim f_i^*(\boldsymbol{x}')\right\}, \tag{54}$$

where $f_i^*(\boldsymbol{x}) \not\precsim f_i^*(\boldsymbol{x}')$ means $f_i^*(\boldsymbol{x}) \npreceq f_i^*(\boldsymbol{x}')$ and $f_i^*(\boldsymbol{x}) \nsucceq f_i^*(\boldsymbol{x}')$, and $\boldsymbol{x} \wedge \boldsymbol{x}'$ is the longest common prefix of $\boldsymbol{x}$ and $\boldsymbol{x}'$, that is, the longest sequence $\boldsymbol{z}$ such that $\boldsymbol{z} \preceq \boldsymbol{x}$ and $\boldsymbol{z} \preceq \boldsymbol{x}'$.

Example 11: Table VI shows $l_{F,i}, i \in [F]$ for the code-tuples $F^{(\beta)}, F^{(\gamma)}$, and $F^{(\delta)}$ in Table III.

Note that from (54) and (27), we obtain

$$l_{F,i} = 0 \Leftrightarrow \mathcal{P}_{F,i} = \{0, 1\} \Leftrightarrow d_{F,i} = 0. \tag{55}$$

TABLE VI
$l_{F,i}, i \in [F]$ FOR THE CODE-TUPLES $F^{(\beta)}, F^{(\gamma)}$ AND $F^{(\delta)}$ IN TABLE III

F	$l_{F,0}$	$l_{F,1}$	$l_{F,2}$
$F^{(\beta)}$	1	2	1
$F^{(\gamma)}$	0	1	0
$F^{(\delta)}$	0	0	0

The following Lemma 12 guarantees that the right hand side of (54) is well-defined.

Lemma 12: For any integer $k \geq 0$, $F(f, \tau) \in \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{ext}}$, and $i \in [F]$, there exist $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$ such that $f_i^*(\boldsymbol{x}) \not\precsim f_i^*(\boldsymbol{x}')$.

Proof of Lemma 12: We prove by contradiction assuming that there exist an integer $k \geq 0$, $F(f, \tau) \in \mathscr{F}_{k\text{-dec}} \cap \mathscr{F}_{\mathrm{ext}}$, and $i \in [F]$ such that for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$, we have $f_i^*(\boldsymbol{x}) \preceq f_i^*(\boldsymbol{x}')$ or $f_i^*(\boldsymbol{x}) \succeq f_i^*(\boldsymbol{x}')$.

Choose two distinct symbols $s, s' \in \mathcal{S}$ arbitrarily and assume $f_i^*(s) \preceq f_i^*(s')$ without loss of generality. From $F \in \mathscr{F}_{\mathrm{ext}}$, we can choose $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$ such that

$$|f_i^*(s\boldsymbol{x})| = |f_i^*(s'\boldsymbol{x}')| \geq |f_i^*(s)| + k. \tag{56}$$

Then, from the assumption, we have

$$f_i^*(s\boldsymbol{x}) \preceq f_i^*(s'\boldsymbol{x}') \text{ or } f_i^*(s\boldsymbol{x}) \succeq f_i^*(s'\boldsymbol{x}'). \tag{57}$$

From (56) and (57),

$$f_i^*(s\boldsymbol{x}) = f_i^*(s'\boldsymbol{x}') \succeq f_i^*(s)\boldsymbol{c}, \tag{58}$$

for some $\boldsymbol{c} \in \mathcal{C}^k$. From (58) and $s \preceq s\boldsymbol{x}$, the pair $(s, \boldsymbol{c}) \in \mathcal{S}^1 \times \mathcal{C}^k$ is not f_i^*-negative. Also, from (58) and $s \npreceq s'\boldsymbol{x}$, the pair $(s, \boldsymbol{c})$ is not f_i^*-positive. Consequently, $(s, \boldsymbol{c}) \in \mathcal{S}^1 \times \mathcal{C}^k \subset \mathcal{S}^* \times \mathcal{C}^k$ is neither f_i^*-positive nor f_i^*-negative. This conflicts with $F \in \mathscr{F}_{k\text{-dec}}$. ∎

Now we state the proof of Lemma 11 as follows.

Proof of Lemma 11: For non-negative integer $t = 0, 1, 2, \ldots$, we define $F^{(t)}(f^{(t)}, \tau^{(t)}) \in \mathscr{F}$ as follows.

$$F^{(t)} := \begin{cases} F & \text{if } t = 0, \\ \widehat{F^{(t-1)}} & \text{if } t \geq 1, \end{cases} \tag{59}$$

that is, $F^{(t)}$ is the code-tuple obtained by applying t times rotation to F.

From $F^{(0)} = F \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}}$ and Lemma 10, for any $t \geq 0$, we have $F^{(t)} \in \mathscr{F}_{\mathrm{reg}} \cap \mathscr{F}_{\mathrm{ext}} \cap \mathscr{F}_{k\text{-dec}}$ and $L(F^{(t)}) = L(F)$. Therefore, to prove Lemma 11, it suffices to prove that there exists an integer $\bar{t} \geq 0$ such that $F^{(\bar{t})} \in \mathscr{F}_{\mathrm{fork}}$. Furthermore, from (55), it suffices to prove that for some integer $\bar{t} \geq 0$, we have $l_{F^{(\bar{t})},0} = l_{F^{(\bar{t})},1} = \cdots = l_{F^{(\bar{t})},|F|-1} = 0$.

Fix $i \in [F]$ and choose $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$ such that $f_i^{*(t)}(\boldsymbol{x}) \not\precsim f_i^{*(t)}(\boldsymbol{x}')$ and $|f_i^{*(t)}(\boldsymbol{x}) \wedge f_i^{*(t)}(\boldsymbol{x}')| = l_{F^{(t)},i}$.

Then we have $f_i^{*(t)}(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})} \not\precsim f_i^{*(t)}(\boldsymbol{x}')d_{F,\tau_i^*(\boldsymbol{x})}$. From Lemma 5, we have $d_{F,i}f_i^{*(t+1)}(\boldsymbol{x}) \not\precsim d_{F,i}f_i^{*(t+1)}(\boldsymbol{x}')$. Hence, we obtain $f_i^{*(t+1)}(\boldsymbol{x}) \not\precsim f_i^{*(t+1)}(\boldsymbol{x}')$. Therefore, it holds that

$$l_{F^{(t+1)},i} \leq |f_i^{*(t+1)}(\boldsymbol{x}) \wedge f_i^{*(t+1)}(\boldsymbol{x}')|. \tag{60}$$

Also, from $f_i^{*(t)}(\boldsymbol{x}) \not\succeq f_i^{*(t)}(\boldsymbol{x}')$ and $|f_i^{*(t)}(\boldsymbol{x}) \wedge f_i^{*(t)}(\boldsymbol{x}')| = l_{F^{(t)},i}$, we have

$$l_{F^{(t)},i} = \left|f_i^{*(t)}(\boldsymbol{x}) \wedge f_i^{*(t)}(\boldsymbol{x}')\right| \tag{61}$$

$$= \left|\left(f_i^{*(t)}(\boldsymbol{x}) d_{F^{(t)},\tau_i^*(\boldsymbol{x})}\right) \wedge \left(f_i^{*(t)}(\boldsymbol{x}') d_{F^{(t)},\tau_i^*(\boldsymbol{x}')}\right)\right|. \tag{62}$$

From Lemma 5, we have

$$l_{F^{(t)},i} = \left|\left(d_{F^{(t)},i} f_i^{*(t+1)}(\boldsymbol{x})\right) \wedge \left(d_{F^{(t)},i} f_i^{*(t+1)}(\boldsymbol{x}')\right)\right| \tag{63}$$

$$= |d_{F^{(t)},i}| + \left|f_i^{*(t+1)}(\boldsymbol{x}) \wedge f_i^{*(t+1)}(\boldsymbol{x}')\right| \tag{64}$$

$$\geq |d_{F^{(t)},i}| + l_{F^{(t+1)},i}, \tag{65}$$

where the inequality is from (60). From (55) and (65), we have $l_{F^{(t+1)},i} = 0$ if $l_{F^{(t)},i} = 0$ and $l_{F^{(t+1)},i} < l_{F^{(t)},i}$ if $l_{F^{(t)},i} > 0$. Therefore, $l_{F^{(t)},i} = 0$ for $t \geq l_{F^{(0)},i}$. Consequently, $F^{(\bar{t})} \in \mathscr{F}_{\text{fork}}$, where $\bar{t} := \max\{l_{F,0}, l_{F,1}, \ldots, l_{F,|F|-1}\}$. ∎

Now we prove the following Lemma 13 that any $F \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{1-dec}}$ can be replaced with some $F' \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$.

Lemma 13: For any $F \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{1-dec}}$, there exists $F' \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$ such that $L(F') = L(F)$.

Proof of Lemma 13: From Lemma 11, we can choose $F'(f', \tau') \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{1-dec}} \cap \mathscr{F}_{\text{fork}}$ such that $L(F') = L(F)$. We prove $F' \in \mathscr{F}_{\text{0-dec}}$ by showing that for $i \in [F]$ and $\boldsymbol{x} \in \mathcal{S}^*$, the pair $(\boldsymbol{x}, \lambda)$ is f'^*_i-positive, that is, for any $i \in [F]$ and $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$ such that $f'^*_i(\boldsymbol{x}) \preceq f'^*_i(\boldsymbol{x}')$, we have $\boldsymbol{x} \preceq \boldsymbol{x}'$.

Choose $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$ such that $f'^*_i(\boldsymbol{x}) \preceq f'^*_i(\boldsymbol{x}')$ arbitrarily. Since $F' \in \mathscr{F}_{\text{fork}}$, we have $\mathcal{P}_{F',\tau'^*_i(\boldsymbol{x})} = \{0, 1\}$, that is, there exist $\boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathcal{S}^*$ such that $f'^*_{\tau'^*_i(\boldsymbol{x})}(\boldsymbol{y}_0) \succeq 0$ and $f'^*_{\tau'^*_i(\boldsymbol{x})}(\boldsymbol{y}_1) \succeq 1$. Hence, we have $f'^*_i(\boldsymbol{x}\boldsymbol{y}_0) = f'^*_i(\boldsymbol{x}) f'^*_{\tau'^*_i(\boldsymbol{x})}(\boldsymbol{y}_0) \succeq f'^*_i(\boldsymbol{x})0$ and $f'^*_i(\boldsymbol{x}\boldsymbol{y}_1) = f'^*_i(\boldsymbol{x}) f'^*_{\tau'^*_i(\boldsymbol{x})}(\boldsymbol{y}_1) \succeq f'^*_i(\boldsymbol{x})1$, where the equalities are from Lemma 1 (i). Since $\boldsymbol{x} \preceq \boldsymbol{x}\boldsymbol{y}_0, \boldsymbol{x} \preceq \boldsymbol{x}\boldsymbol{y}_1$, and $F' \in \mathscr{F}_{\text{1-dec}}$, the pairs $(\boldsymbol{x}, 0), (\boldsymbol{x}, 1)$ are f'^*_i-positive.

From $f'^*_i(\boldsymbol{x}) \preceq f'^*_i(\boldsymbol{x}')$ and $F' \in \mathscr{F}_{\text{ext}}$, there exist $c, c' \in \mathcal{C}^1$ and $\boldsymbol{x}'' \in \mathcal{S}^*$ such that $f'^*_i(\boldsymbol{x})c \preceq f'^*_i(\boldsymbol{x}')c' \preceq f'^*_i(\boldsymbol{x}'\boldsymbol{x}'')$. Since $(\boldsymbol{x}, 0)$ and $(\boldsymbol{x}, 1)$ are f'^*_i-positive, we have $\boldsymbol{x} \preceq \boldsymbol{x}'\boldsymbol{x}''$. Therefore, we have either (a) or (b) of the following conditions: (a) $\boldsymbol{x} \preceq \boldsymbol{x}'$, (b) $\boldsymbol{x} \neq \boldsymbol{x}'$ and $\boldsymbol{x} \succeq \boldsymbol{x}'$. To complete the proof, it suffices to prove that (a) is true. Now we prove it by contradiction assuming that (b) is true, that is, there exists $\boldsymbol{z} = z_1 z_2 \ldots z_{|\boldsymbol{z}|} \in \mathcal{S}^* \setminus \{\lambda\}$ such that $\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{z}$.

From $\boldsymbol{x} \succeq \boldsymbol{x}'$, Lemma 1 (iii), and $f'^*_i(\boldsymbol{x}) \preceq f'^*_i(\boldsymbol{x}')$, we have

$$f'^*_i(\boldsymbol{x}') = f'^*_i(\boldsymbol{x}). \tag{66}$$

Choose $s \in \mathcal{S} \setminus \{z_{|\boldsymbol{z}|}\}$ and define $\boldsymbol{z}' = z_1 z_2 \ldots z_{|\boldsymbol{z}|-1} s$. From $F' \in \mathscr{F}_{\text{ext}}$, we can choose $\boldsymbol{y}' \in \mathcal{S}^*$ and $c' \in \mathcal{C}$ such that

$$f'^*_{\tau'^*_i(\boldsymbol{x}')}(\boldsymbol{z}'\boldsymbol{y}') \succeq c'. \tag{67}$$

From (66) and (67), we have

$$f'^*_i(\boldsymbol{x})c' = f'^*_i(\boldsymbol{x}')c' \preceq f'^*_i(\boldsymbol{x}') f'^*_{\tau'^*_i(\boldsymbol{x}')}(\boldsymbol{z}'\boldsymbol{y}') = f'^*_i(\boldsymbol{x}'\boldsymbol{z}'\boldsymbol{y}'), \tag{68}$$

where the last equality is from Lemma 1 (i). Since $(\boldsymbol{x}, 0)$ and $(\boldsymbol{x}, 1)$ are f'^*_i-positive (in particular, $(\boldsymbol{x}, c')$ is f'^*_i-positive), we have $\boldsymbol{x} \preceq \boldsymbol{x}'\boldsymbol{z}'\boldsymbol{y}'$. From $\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{z}$, we have $\boldsymbol{x}'\boldsymbol{z} \preceq \boldsymbol{x}'\boldsymbol{z}'\boldsymbol{y}'$. Hence, we obtain $\boldsymbol{z} \preceq \boldsymbol{z}'\boldsymbol{y}'$. This conflicts with the definition of $\boldsymbol{z}'$. ∎

Example 12: Consider $F^{(\beta)} \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{1-dec}}$ of Table III. Lemma 13 guarantees that there exists $F' \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$ such that $L(F') = L(F^{(\beta)})$. Indeed, $F^{(\delta)}$ of Table III satisfies $F^{(\delta)} \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$ and $L(F^{(\delta)}) = L(F^{(\beta)})$.

D. Proof of Theorem 1

Finally, we prove the following Theorem 1 as the main result of this paper.

Theorem 1: For any $F(f, \tau) \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{1-dec}}$, we have

$$L(F) \geq L_{\text{Huff}}, \tag{69}$$

where L_{Huff} is the average codeword length of the Huffman code.

Proof of Theorem 1: From Lemma 13, there exists $F' \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$ such that $L(F') = L(F)$. Thus, we can assume $F \in \mathscr{F}_{\text{reg}} \cap \mathscr{F}_{\text{ext}} \cap \mathscr{F}_{\text{0-dec}}$ without loss of generality.

Let $a \in \arg\min_{i \in [F]} L_i(F)$, and define $F'(f'_0, \tau'_0) \in \mathscr{F}^{(1)}$ as $f'_0 := f_a, \tau'_0 := 0$. From $F' \in \mathscr{F}^{(1)}$, the simultaneous equations (20) and (21) have the unique solution $\boldsymbol{\pi}(F') = (\pi_0(F')) = (1)$. Therefore, we have $F' \in \mathscr{F}_{\text{reg}}$ and

$$L(F') = \pi_0(F') L_0(F') = L_a(F)$$
$$= \sum_{i \in [F]} \pi_i(F) L_a(F) \leq \sum_{i \in [F]} \pi_i(F) L_i(F) = L(F), \tag{70}$$

where the inequality is from $a \in \arg\min_{i \in [F]} L_i(F)$.

From $F \in \mathscr{F}_{\text{0-dec}}$ and Lemma 4, for any $i \in [F]$, f_i is prefix-free. In particular, $f'_0 = f_a$ is prefix-free. Therefore, from Lemma 4, we have $F' \in \mathscr{F}_{\text{0-dec}}$. Hence, from Lemma 3 (ii), the code table f'^*_0 is injective, that is, f'^*_0 is uniquely decodable code with a single code table. Therefore, from McMillan's Theorem [2], we have

$$L(F') \geq L_{\text{Huff}}. \tag{71}$$

From (70) and (71), we obtain $L(F) \geq L_{\text{Huff}}$. ∎

IV. Conclusion

We discussed the optimality of Huffman code in the class of 1-bit delay decodable codes with a finite number of code tables. First, we introduced a code-tuple as a model of a time-variant encoder with a finite number of code tables. Next, we define the class of k-bit delay decodable coe-tuples for $k = 0, 1, 2, \ldots$. Then, we proved that Huffman code achieves the optimal average codeword length in the class of 1-bit delay decodable code-tuples in Theorem 1.

We implicitly imposed the constraint that the encoder determines which code table to use following only the current code table and symbol, independently of the past symbols. The question remains whether relaxing this constraint results in a shorter average codeword length than the Huffman code.

Appendix A

Proof of Lemma 9: To state the proof of Lemma 9, first we prove the following Lemma 14.

Lemma 14: For $F(f,\tau) \in \mathscr{F}, i \in [F]$, and $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{S}^*$, if $\boldsymbol{x}' \preceq \boldsymbol{x}$ and $f_i^*(\boldsymbol{x}) = f_i^*(\boldsymbol{x}')$, then $\mathcal{P}_{F,\tau_i^*(\boldsymbol{x}')} \supseteq \mathcal{P}_{F,\tau_i^*(\boldsymbol{x})}$.

Proof of Lemma 14: Let $c \in \mathcal{P}_{F,\tau_i^*(\boldsymbol{x})}$. Then there exists $\boldsymbol{y} \in \mathcal{S}^*$ such that $f^*_{\tau_i^*(\boldsymbol{x})}(\boldsymbol{y}) \succeq c$, and we have $f_i^*(\boldsymbol{x})c \preceq f_i^*(\boldsymbol{x})f^*_{\tau_i^*(\boldsymbol{x})}(\boldsymbol{y})$. From the assumption that $f_i^*(\boldsymbol{x}) = f_i^*(\boldsymbol{x}')$, we have $f_i^*(\boldsymbol{x}')c \preceq f_i^*(\boldsymbol{x})f^*_{\tau_i^*(\boldsymbol{x})}(\boldsymbol{y})$. Let $\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{z}$. Then we have $f_i^*(\boldsymbol{x}')c \preceq f_i^*(\boldsymbol{x}'\boldsymbol{z})f^*_{\tau_i^*(\boldsymbol{x}'\boldsymbol{z})}(\boldsymbol{y})$. From Lemma 1 (i), we have $f_i^*(\boldsymbol{x}')c \preceq f_i^*(\boldsymbol{x}')f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z})f^*_{\tau_i^*(\boldsymbol{x}'\boldsymbol{z})}(\boldsymbol{y})$. Thus, it holds that $c \preceq f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z})f^*_{\tau_i^*(\boldsymbol{x}'\boldsymbol{z})}(\boldsymbol{y}) = f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}\boldsymbol{y})$ from Lemma 1 (i) and Lemma 1 (ii), Hence, we obtain $c \in \mathcal{P}_{F,\tau_i^*(\boldsymbol{x}')}$. ∎

Proof of Lemma 9: Let $(k, i, \boldsymbol{x}, \boldsymbol{x}')$ be a tuple satisfying all of the conditions (i)–(iii), and we lead a contradiction.

From $\boldsymbol{x}' \preceq \boldsymbol{x}$ of the condition (iii) and Lemma 1 (iii), we have

$$f_i^*(\boldsymbol{x}') \preceq f_i^*(\boldsymbol{x}). \tag{72}$$

From the condition (ii) $|\widehat{f_i^*}(\boldsymbol{x})| + k \leq |\widehat{f_i^*}(\boldsymbol{x}')|$, we have $|d_{F,i}| + |\widehat{f_i^*}(\boldsymbol{x})| + k \leq |d_{F,i}| + |\widehat{f_i^*}(\boldsymbol{x}')|$. Thus, $|d_{F,i}\widehat{f_i^*}(\boldsymbol{x})| + k \leq |d_{F,i}\widehat{f_i^*}(\boldsymbol{x}')|$. From Lemma 5, we have $|f_i^*(\boldsymbol{x})d_{F,\tau_i^*(\boldsymbol{x})}| + k \leq |f_i^*(\boldsymbol{x}')d_{F,\tau_i^*(\boldsymbol{x}')}|$. Consequently,

$$\left|f_i^*(\boldsymbol{x}')\right| \geq \left|f_i^*(\boldsymbol{x})\right| + \left|d_{F,\tau_i^*(\boldsymbol{x})}\right| + k - \left|d_{F,\tau_i^*(\boldsymbol{x}')}\right|. \tag{73}$$

From (72),

$$\left|d_{F,\tau_i^*(\boldsymbol{x})}\right| + k - \left|d_{F,\tau_i^*(\boldsymbol{x}')}\right| \leq 0. \tag{74}$$

Since $0 \leq |d_{F,\tau_i^*(\boldsymbol{x})}| \leq 1$ and $0 \leq |d_{F,\tau_i^*(\boldsymbol{x}')}| \leq 1$, the following two cases are possible: (i) $|d_{F,\tau^*(\boldsymbol{x})}| + k - |d_{F,\tau^*(\boldsymbol{x}')}| = 0$ and (ii) $|d_{F,\tau^*(\boldsymbol{x})}| + k - |d_{F,\tau^*(\boldsymbol{x}')}| = -1$.

(i) the case $|d_{F,\tau_i^*(\boldsymbol{x})}| + k - |d_{F,\tau_i^*(\boldsymbol{x}')}| = 0$: From (73),

$$\left|f_i^*(\boldsymbol{x}')\right| \geq \left|f_i^*(\boldsymbol{x})\right|. \tag{75}$$

From (72) and (75),

$$f_i^*(\boldsymbol{x}) = f_i^*(\boldsymbol{x}'). \tag{76}$$

From Lemma 14, $\mathcal{P}_{F,\tau_i^*(\boldsymbol{x}')} \supseteq \mathcal{P}_{F,\tau_i^*(\boldsymbol{x})}$. In particular,

$$\mathcal{P}_{F,\tau_i^*(\boldsymbol{x})} = \{0,1\} \Rightarrow \mathcal{P}_{F,\tau_i^*(\boldsymbol{x}')} = \{0,1\}. \tag{77}$$

From (27),

$$\left|d_{F,\tau_i^*(\boldsymbol{x})}\right| = 0 \Rightarrow \left|d_{F,\tau_i^*(\boldsymbol{x}')}\right| = 0. \tag{78}$$

From (78) and $|d_{F,\tau_i^*(\boldsymbol{x})}| + k - |d_{F,\tau_i^*(\boldsymbol{x}')}| = 0$, we have $k = 0$. Hence, from the condition (i), we obtain $F \in \mathscr{F}_{0\text{-dec}}$. Therefore, from (76) and Lemma 3 (ii), we obtain $\boldsymbol{x} = \boldsymbol{x}'$, which conflicts with $\boldsymbol{x} \neq \boldsymbol{x}'$ of the condition (ii).

(ii) the case $|d_{F,\tau_i^*(\boldsymbol{x})}| + k - |d_{F,\tau_i^*(\boldsymbol{x}')}| = -1$: We have $|d_{F,\tau_i^*(\boldsymbol{x})}| = k = 0$ and $|d_{F,\tau_i^*(\boldsymbol{x}')}| = 1$. From the condition (i) and $k = 0$, we have

$$F \in \mathscr{F}_{0\text{-dec}}. \tag{79}$$

From (72),

$$\left|f_i^*(\boldsymbol{x}')\right| \leq \left|f_i^*(\boldsymbol{x})\right|. \tag{80}$$

From (73) and $|d_{F,\tau_i^*(\boldsymbol{x})}| + k - |d_{F,\tau_i^*(\boldsymbol{x}')}| = -1$,

$$\left|f_i^*(\boldsymbol{x})\right| \leq \left|f_i^*(\boldsymbol{x}')\right| + 1. \tag{81}$$

From (80) and (81), we have either $|f_i^*(\boldsymbol{x})| = |f_i^*(\boldsymbol{x}')|$ or $|f_i^*(\boldsymbol{x})| + 1 = |f_i^*(\boldsymbol{x}')|$. If we assume $|f_i^*(\boldsymbol{x})| = |f_i^*(\boldsymbol{x}')|$, then $f_i^*(\boldsymbol{x}) = f_i^*(\boldsymbol{x}')$ holds from (72). Then from (79) and Lemma 3 (ii), we obtain $\boldsymbol{x} = \boldsymbol{x}'$, which conflicts with $\boldsymbol{x} \neq \boldsymbol{x}'$ of the condition (ii). Hence, we have

$$\left|f_i^*(\boldsymbol{x})\right| = \left|f_i^*(\boldsymbol{x}')\right| + 1. \tag{82}$$

From the condition (ii) $\boldsymbol{x}' \preceq \boldsymbol{x}$ and $\boldsymbol{x} \neq \boldsymbol{x}'$, there exists $\boldsymbol{z} = z_1 z_2 \dots z_{|\boldsymbol{z}|} \in \mathcal{S}^* \backslash \{\lambda\}$ such that $\boldsymbol{x} = \boldsymbol{x}'\boldsymbol{z}$. For such $\boldsymbol{z}$, from (82), we have $|f_i^*(\boldsymbol{x}'\boldsymbol{z})| = |f_i^*(\boldsymbol{x}')| + 1$. Then from Lemma 1 (i), we have $|f_i^*(\boldsymbol{x}')| + |f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z})| = |f_i^*(\boldsymbol{x}')| + 1$. Thus, we obtain

$$\left|f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z})\right| = 1. \tag{83}$$

Choose $s \in \mathcal{S}\backslash\{z_{|\boldsymbol{z}|}\}$ and define $\boldsymbol{z}' := z_1 z_2 \dots z_{|\boldsymbol{z}|-1} s$. From $F \in \mathscr{F}_{\text{ext}}$, we can choose $\boldsymbol{y} \in \mathcal{S}^*$ such that

$$\left|f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}'\boldsymbol{y})\right| \geq 1. \tag{84}$$

From $|d_{F,\tau_i^*(\boldsymbol{x}')}| = 1$ (i.e., $\mathcal{P}_{F,\tau_i^*(\boldsymbol{x}')} = \{d_{F,\tau_i^*(\boldsymbol{x}')}\}$), (83) and (84), we have

$$f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}) = d_{F,\tau_i^*(\boldsymbol{x}')}, \tag{85}$$

and

$$f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}'\boldsymbol{y}) \succeq d_{F,\tau_i^*(\boldsymbol{x}')}. \tag{86}$$

Therefore,

$$\begin{aligned} f_i^*(\boldsymbol{x}'\boldsymbol{z}) &= f_i^*(\boldsymbol{x}')f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}) = f_i^*(\boldsymbol{x}')d_{F,\tau_i^*(\boldsymbol{x}')} \\ &\preceq f_i^*(\boldsymbol{x}')f^*_{\tau_i^*(\boldsymbol{x}')}(\boldsymbol{z}'\boldsymbol{y}) = f_i^*(\boldsymbol{x}'\boldsymbol{z}'\boldsymbol{y}), \end{aligned} \tag{87}$$

where the first equality is from Lemma 1 (i), the second equality is from (85), the prefix relation is from (86), and the last equality is from Lemma 1 (i). From (79) and Lemma 3 (i), we have $\boldsymbol{x}'\boldsymbol{z} \preceq \boldsymbol{x}'\boldsymbol{z}'\boldsymbol{y}$. Thus, we obtain $\boldsymbol{z} \preceq \boldsymbol{z}'\boldsymbol{y}$. This conflicts with the definition of $\boldsymbol{z}'$. ∎

Appendix B

List of Notations

$\mathcal{A} \times \mathcal{B}$	$\{(a,b) : a \in \mathcal{A}, b \in \mathcal{B}\}$, defined at the beginning of Section II.
$\lvert\mathcal{A}\rvert$	the cardinality of a set $\mathcal{A}$, defined at the beginning of Section II.
$\mathcal{A}^k$	the set of all sequences of length k over a set $\mathcal{A}$, defined at the beginning of Section II.
$\mathcal{A}^*$	the set of all sequences of finite length over a set $\mathcal{A}$, defined at the beginning of Section II.
$\mathcal{C}$	the coding alphabet $\mathcal{C} = \{0,1\}$, defined in Section I.
$d_{F,i}$	defined in (27).
f_i^*	defined in (5).
F	simplified notation of a code-tuple $F(f_i, \tau_i : i \in [m])$, also written as $F(f,\tau)$, defined below Definition 1.
$\lvert F\rvert$	the number of code tables of F, defined below Definition 1.
$[F]$	simplified notation of $[\lvert F\rvert] = \{0,1,2,\dots,\lvert F\rvert-1\}$, defined below Definition 1.

$\widehat{F}$	the code-tuple obtained by applying rotation to F, defined in Definition 10.
$\mathscr{F}^{(m)}$	the set of all m-code-tuples, defined after Definition 1.
$\mathscr{F}$	the set of all code-tuples, defined after Definition 1.
$\mathscr{F}_{\mathrm{ext}}$	defined in Definition 3.
$\mathscr{F}_{\mathrm{fork}}$	defined in Definition 11.
$\mathscr{F}_{k\text{-dec}}$	the set of all k-bit delay decodable code-tuples, defined in Definition 5.
$\mathscr{F}_{\mathrm{reg}}$	the set of all regular code-tuples, defined in Definition 7.
$l_{F,i}$	defined in (54).
$L(F)$	the average codeword length of a code-tuple F, defined in Definition 8.
$L_i(F)$	the average codeword length of the i-th code table of F, defined in Definition 8.
$[m]$	$\{0, 1, 2, \ldots, m-1\}$, defined at the beginning of Section II-A.
$\mathcal{P}_{F,i}$	the set of all $c \in \mathcal{C}$ which is the first bit of $f_i^*(\boldsymbol{x})$ for some $\boldsymbol{x} \in \mathcal{S}^*$, defined in Definition 9.
$Q(F)$	the transition probability matrix, defined in Definition 6.
$Q_{i,j}(F)$	the transition probability, defined in Definition 6.
$\mathcal{S}$	the source alphabet, defined at the beginning of Section I.
$\boldsymbol{x} \wedge \boldsymbol{y}$	the longest common prefix of $\boldsymbol{x}$ and $\boldsymbol{y}$, defined after Lemma 11.
$\boldsymbol{x} \preceq \boldsymbol{y}$	$\mathbf{x}$ is a prefix of $\mathbf{y}$, defined at the beginning of Section II.
$\boldsymbol{x} \not\succeq\not\preceq \boldsymbol{y}$	$\mathbf{x}$ is not a prefix of $\mathbf{y}$, and $\mathbf{y}$ is not a prefix of $\mathbf{x}$, defined after Lemma 11.
$\mathrm{suf}(\boldsymbol{x})$	the sequence obtained by deleting the first letter of $\boldsymbol{x}$, defined at the beginning of Section II.
$\lvert\boldsymbol{x}\rvert$	the length of a sequence $\boldsymbol{x}$, defined at the beginning of Section II.
λ	the empty sequence, defined at the beginning of Section II.
$\mu(s)$	the probability of occurrence of symbol s, defined at the beginning of Section II-C.
$\boldsymbol{\pi}(F)$	defined in Definition 7.
σ	the alphabet size, defined at the beginning of Section I.
τ_i^*	defined in (6).

ACKNOWLEDGMENT

The authors would like to thank the editor and anonymous reviewers for their valuable comments and suggestions. The authors would also like to thank Hirosuke Yamamoto and Hiroshi Fujisaki for comments on an early draft.

REFERENCES

[1] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proc. IRE*, vol. 40, no. 9, pp. 1098–1102, Sep. 1952.

[2] B. McMillan, "Two inequalities implied by unique decipherability," *IRE Trans. Inf. Theory*, vol. 2, no. 4, pp. 115–116, Dec. 1956.

[3] H. Yamamoto, M. Tsuchihashi, and J. Honda, "Almost instantaneous fixed-to-variable length codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 12, pp. 6432–6443, Dec. 2015.

[4] K. Hashimoto and K. Iwata, "On the optimality of binary AIFV codes with two code trees," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Melbourne, VIC, Australia, Jul. 2021, pp. 3173–3178.

[5] W. Hu, H. Yamamoto, and J. Honda, "Worst-case redundancy of optimal binary AIFV codes and their extended codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 8, pp. 5074–5086, Aug. 2017.

[6] R. Fujita K. Iwata, and H. Yamamoto, "Worst-case redundancy of optimal binary AIFV-m codes for $m = 3, 5$," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020, pp. 2355–2359.

[7] K. Iwata, and H. Yamamoto, "A dynamic programming algorithm to construct optimal code trees of AIFV codes," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Monterey, CA, USA, Oct./Nov. 2016, pp. 641–645.

[8] K. Iwata, and H. Yamamoto, "An iterative algorithm to construct optimal binary AIFV-m codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Kaohsiung, Taiwan, Oct. 2017, pp. 519–523.

[9] K. Sumigawa and H. Yamamoto, "Coding of binary AIFV code trees," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 1152–1156.

[10] T. Hiraoka and H. Yamamoto, "Alphabetic AIFV codes constructed from Hu-Tucker codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 2182–2186.

[11] R. Fujita, K. Iwata, and H. Yamamoto, "An optimality proof of the iterative algorithm for AIFV-m codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 2187–2191.

[12] T. Hiraoka and H. Yamamoto, "Dynamic AIFV coding," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Singapore, Oct. 2018, pp. 577–581.

[13] R. Fujita, K. Iwata, and H. Yamamoto, "An iterative algorithm to optimize the average performance of markov chains with finite states," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1902–1906.

[14] K. Hashimoto, K. Iwata, and H. Yamamoto, "Enumeration and coding of compact code trees for binary AIFV codes," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1527–1531.

[15] M. J. Golin and E. Y. Harb, "Polynomial time algorithms for constructing optimal AIFV codes," in *Proc. Data Compression Conf. (DCC)*, Snowbird, UT, USA, Mar. 2019, pp. 231–240.

[16] M. J. Golin and E. Y. Harb, "Polynomial time algorithms for constructing optimal binary AIFV-2 codes," 2020, *arXiv:2001.11170*.

[17] H. Yamamoto, K. Imaeda, K. Hashimoto, and K. Iwata, "A universal data compression scheme based on the AIFV coding techniques," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, LA, USA, Jun. 2020, pp. 2378–2382.

[18] K. Iwata and H. Yamamoto, "An algorithm for constructing the optimal code trees for binary alphabetic AIFV-m codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Riva de Garda, Italy, Apr. 2021, pp. 261–265.

[19] M. J. Golin and E. Y. Harb, "Speeding up the AIFV-2 dynamic programs by two orders of magnitude using range minimum queries," *Theor. Comput. Sci.*, vol. 865, no. 14, pp. 99–118, Apr. 2021.

[20] M. J. Golin and A. J. L. Parupat, "Speeding Up AIFV-m dynamic programs by $m-1$ orders of magnitude," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Espoo, Finland, Jun./Jul. 2022, pp. 282–287.

[21] R. Sugiura, Y. Kamamoto, N. Harada, and T. Moriya, "Optimal Golomb-Rice code extension for lossless coding of low-entropy exponentially distributed sources," *IEEE Trans Inf. Theory*, vol. 64, no. 4, pp. 3153–3161, Apr. 2018.

[22] R. Sugiura, Y. Kamamoto, and T. Moriya, "General form of almost instantaneous fixed-to-variable-length codes and optimal code tree construction," 2022, *arXiv:2203.08437*.

Kengo Hashimoto (Graduate Student Member, IEEE) was born in Japan, in 1994. He received the B.E. and M.E. degrees from the Department of Information Science, University of Fukui, in 2019 and 2021, respectively, where he is currently pursuing the Ph.D. degree. His research interests include information theory and coding theory.

Ken-ichi Iwata (Member, IEEE) received the B.Ed. degree from Wakayama University in 1993, the M.Sc. degree in information science from the Japan Advanced Institute of Science and Technology in 1995, and the D.E. degree from the University of Electro-Communications in 2006. Since 2008, he has been an Associate Professor with the Graduate School of Engineering, University of Fukui.