IEICE TRANS. FUNDAMETALS, VOL. E81–A, NO. 1 JANUARY 1998

PAPER Special Section on Cryptography and Information Security

Window and Extended Window Methods for Addition Chain and Addition-Subtraction Chain

Noboru KUNIHIRO† *and* Hirosuke YAMAMOTO††, *Members*

SUMMARY The addition chain (A-chain) and addition-subtraction chain (AS-chain) are efficient tools to calculate power M^e (or multiplication eM), where integer e is fixed and M is variable. Since the optimization problem to find the shortest A (or AS)-chain is NP-hard, many algorithms to get a sub-optimal A (or AS)-chain in polynomial time are proposed. In this paper, a window method for the AS-chain and an extended window method for the A-chain and AS-chain are proposed and their performances are theoretically evaluated by applying the theory of the optimal variable-to-fixed length code, i.e., Tunstall code, in data compression. It is shown by theory and simulation that the proposed algorithms are more efficient than other algorithms in practical cases in addition to the asymptotic case.

key words: *addition-subtraction chain, Tunstall code, canonical signed binary representation, window method*

1. Introduction

Calculation of powers plays a main role in several cryptosystems, prime testing algorithms, or integer factoring algorithms. For instance, the enciphering (and deciphering) of RSA scheme is given by $M^e \bmod n$, where M is a plaintext (ciphertext) and e is a public (secret) key. Since the key length is usually more than 500 bits, we need an efficient algorithm to calculate the powers. The fast calculation can be realized by reducing the number of multiplication as small as possible and/or speeding up the operation of multiplication. In this paper, we consider the former problem, i.e., how to reduce the number of multiplication.

It is well known that the powers can be calculated efficiently by using the addition chain [1]. Suppose that M^e can be calculated like $M^1 \to M^{a_1} \to M^{a_2} \to \cdots \to M^{a_r}(= M^e)$ based on a rule:

$$M^{a_i} = M^{a_j} \times M^{a_k}, \quad j, k < i. \tag{1}$$

Then the sequence of exponents a_i, $\langle a_0(=1), a_1, a_2, \cdots, a_r(=e)\rangle$, is called an addition chain since this sequence satisfies the relation

$$a_i = a_j + a_k, \quad j, k < i. \tag{2}$$

Our main purpose is to find an efficient algorithm to derive a short addition chain for a given e. Since the addition chain does not depend on M, the calculation based on the addition chain is efficient when M is variable and e is fixed*.

The addition chain (A-chain) is applicable not only to the power operation but also to arbitrary group operations. Recently, several cryptosystems based on the Abelian group defined over elliptic curves are proposed, in which an inverse element can easily be obtained. Hence, for such groups, the computation rule can be extended to

$$a_i = a_j \pm a_k, \quad j, k < i. \tag{3}$$

The chain satisfying this rule is called an addition-subtraction chain (AS-chain). The A-chain can be considered as a special case of the AS-chain, but we use A-AS-chains to stand for both the A-chain and AS-chain.

Since obtaining the shortest A-AS-chains is NP-hard [3], restricted A-AS-chains, e.g., the star chain [1], are considered or some algorithms are proposed to attain shorter A-AS-chains [1],[4]–[7]. However, since such algorithms are heuristic, it is not known how they are close to the optimal even in some restricted class of A-AS-chains.

In this paper, we propose a window method for the AS-chain. We also propose the extended window method, for the A-AS-chains, which generates a special case of the star chain, but it includes many known methods, e.g. the binary method [1], the window method for the A-chain [1], Morain-Olivos (MO) method [5], Koyama-Tsuruoka (KT) method [7] and our window method for the AS-chain as special cases. For the extended window method, we derive a theoretical lower bound of the chain length by invoking the source coding theorem for variable-to-fixed length codes. This coding theorem also brings us efficient algorithms based on the Tunstall code, which are optimal in the sense that they can attain the above lower bound asymptotically. Furthermore, it is shown that the proposed methods are more efficient than known methods [1], [4]–[7] in practical cases.

In this paper, the base of logarithm is 2. We use notation $\bar{1} = -1$ to represent signed binary numbers. Note $\bar{\bar{1}} = 1, \bar{0} = 0$, etc. Furthermore, letting

Manuscript received March 27, 1997.
Manuscript revised July 12, 1997.

†The author is with NTT Communication Science Laboratories, Kyoto-fu, 619–02 Japan.
††The author is with the Faculty of Engineering, the University of Tokyo, Tokyo, 113 Japan.

*Exponentiation M^x where M is fixed and x is variable can be efficiently calculated by other methods. Refer, e.g., [2].

$Seq = s_0 s_1 \dots s_{L-1}, s_l \in \{\bar{1}, 0, 1\}$, be a signed binary sequence, then $|Seq|$ stands for the length of Seq, i.e. L, and $\overline{Seq} = \bar{s}_0 \bar{s}_1 \dots \bar{s}_{L-1}$. Although the ordinary binary representation of a given number is unique, the signed binary representation is not unique, which may be given by, e.g., Booth algorithm [8], [9], Koyama–Tsuruoka algorithm [7], etc. For instance, $(23)_{10} = 10111 = 10\bar{1}00\bar{1}$. Note that Booth algorithm generates the canonical signed binary representation (CSBR) that are *sparse* and *minimal*, i.e., no nonzero bits are adjacent and the number of nonzero bits is minimal.

We assume for simplicity that letting Seq_e be the binary representation of e, each bit s_l of Seq_e occurs with $p = \Pr\{s_l = 0\}$ and $q = 1 - p = \Pr\{s_l = 1\}$ except the most significant bit (MSB)†. The MSB of Seq_e is always one. We represent the bit length of the exponent e by $d = |Seq_e| = \lfloor \log e \rfloor + 1$.

2. Window Method for the Ordinary Binary Representation

For a given e, a simple addition chain can be obtained by the so-called *binary method* [1], in which a_i is repeatedly doubled by $a_i = 2a_{i-1}$ initially from $a_0 = 1$ until a_i satisfies $e/2 < a_i \leqq e$, and each a_{i-1} is added if the i-th bit is one in the binary representation of e. For instance, $e = 50 = (110010)_2$ can be obtained by an A-chain $\langle 1, 2, 4, 8, 16, 32, 48(= 32 + 16), 50(= 48 + 2) \rangle$. However, the binary method is not efficient, especially when e is a little less than 2^ℓ. On the other hand, it is known that obtaining the shortest A-chain is NP-hard [3] for the general addition rule given by Eq. (2). Hence, several restricted A-chains have been considered.

If j is restricted to $i - 1$ in Eq. (2), the chain is called *star chain* and its property is studied in [1]. But, since it is still difficult to derive the optimal star chain, we must narrow the addition rule furthermore.

In the *window method* [1], the addition is restricted to the following two rules.

$$a_i = a_{i-1} + a_{i-1} = 2a_{i-1} \tag{4}$$

$$a_i = a_{i-1} + a_k, \quad a_k \in \mathcal{D}_\kappa, \tag{5}$$

where $\mathcal{D}_\kappa$ is a set of integers that have length κ in the binary representation with MSB= 1. Integer κ is called window length. The first rule is the *doubling rule* like the binary method while the second is the star chain rule, but a_k must be in $\mathcal{D}_\kappa$. In the window method, the shortest addition sequence (A-sequence)†† is first calculated for integers included in $\mathcal{D}_\kappa$, and the A-chain for e is constructed by applying (4) and (5) to the binary representation of e, Seq_e. For instance, in case of $\kappa = 3$ and $e = 172$, we have $\mathcal{D}_\kappa = \{4, 5, 6, 7\}$ and the A-sequence for $\mathcal{D}_\kappa$ is given by $\langle 1, 2, 4, 5, 6, 7 \rangle$. Since

$$Seq_e = \underbrace{101}_{5}\,0\,\underbrace{110}_{6}\,0,$$

the remaining A-chain becomes $\langle 5, 10, 20, 40, 80, 86(= 80 + 6), 172 \rangle$.

We note that if $a_k = 2\tilde{a}_k$, i.e., a_k is even, then $2a_{i-1} + a_k = 2(a_{i-1} + \tilde{a}_k)$ holds. This means that, by exchanging the applying order of the doubling and star chain rules, we can use odd number $\tilde{a}_k$ instead of even number a_k in (5), i.e., $\tilde{\mathcal{D}}_\kappa$, which consists of odd integers less than 2^κ, instead of $\mathcal{D}_\kappa$. For the above example, we can use $\tilde{\mathcal{D}}_\kappa = \{1, 3, 5, 7\}$ with A-sequence $\langle 1, 2, 3, 5, 7 \rangle$, and the remaining A-chain for e is given by $\langle 5, 10, 20, 40, 43(= 40 + 3), 86, 172 \rangle$ because of

$$Seq_e = \underbrace{101}_{5}\,0\,\underbrace{11}_{3}\,00.$$

Since the A-sequence of $\tilde{\mathcal{D}}_\kappa$ is usually shorter than the one of $\mathcal{D}_\kappa$, we should use $\tilde{\mathcal{D}}_\kappa$ rather than $\mathcal{D}_\kappa$.

When $\kappa = 1$ in the window method, then, for instance, A-chain of $e = 50 = (110010)_2$ is obtained as $\langle 1, 2, 3(= 2 + 1), 6, 12, 24, 25(= 24 + 1), 50 \rangle$. The length of this A-chain is equal to the one derived by the binary method†††. Hence, the binary method can be considered as a special case of the window method.

The average chain length l_{WB} for the window method is given as follows.

Theorem 1: The window method for the ordinary binary representation Seq_e satisfies

$$l_{WB} \approx d - \left(\kappa - \frac{p - p^\kappa}{1 - p}\right) + \frac{d}{\kappa + \frac{p}{1-p}} + 2^{\kappa - 1}, \tag{6}$$

where $d = \lfloor \log e \rfloor + 1$ and κ is the window length††††.

Proof: The proof is shown in [10]†††††. □

3. Window Method for CSBR

The binary method can be applied for the AS-chain. Actually MO method [5] is the binary method for the CSBR. We now show that the window method can also be applied for the CSBR.

Window method for CSBR

1. Obtain Seq_e with CSBR by applying Booth Algorithm to e.
2. Set the window length κ.
3. Construct the A-sequence $\langle 1, 2, 3, 5, \cdots, \frac{2}{3}(2^\kappa -$

†Practically, p and q may be determined from the frequencies of 0 and 1 in Seq_e.

††In the A-chain, a given number occurs at the end. On the other hand, in the A-sequence, given numbers occur in the sequence.

†††Refer [1] for the details of this equivalence.

††††Although Eq. (6) does not hold with equality exactly, the right side is very tight approximation of l_{WB}.

†††††Equation (6) can also be derived from Eq. (11)–(13).

Table 1 Performance of proposed method and theoretical bounds.

	A-Chain				AS-Chain with CSBR			
	Window Method		Lower Bound		Window Method		Lower Bound	
bit	l_{WB}	$K+1$	l_{EB}	$K+1$	l_{WS}	$K+1$	l_{ES}	$K+1$
512	609.3	$16(\kappa=5)$	607.7	20	598.9	$21(\kappa=6)$	594.7	16
1024	1197.3	$32(\kappa=6)$	1196.3	32	1181.3	$21(\kappa=6)$	1175.4	26

$(-1)^\kappa) - 1\rangle$, which contains all odd numbers less than $\frac{2}{3}(2^\kappa - (-1)^\kappa)$.

4. The sequence Seq_e is processed from the MSB in the following two phases. We call "phase 1 + phase 2" one cycle.

   Phase1 Read κ bits from Seq_e. Let the κ bits be $s_0 s_1 \ldots s_{\kappa-l-1} \underbrace{0\cdots0}_{l}$ where $s_{\kappa-l-1} \neq 0$, and $(s_0 s_1 \ldots s_{\kappa-l-1})_{10} = \tilde{a}$. Note that $\tilde{a}$ is positive if $s_0 = 1$ or negative if $s_0 = \bar{1}$. First apply the doubling rule $\kappa - l$ times. Next apply the star chain rule, $a_i = a_{i-1} + \tilde{a}$, and finally apply the doubling rule l times. Remove the κ bits from Seq_e.

   Phase2 If $Seq_e = \underbrace{0\cdots0}_{l} s_l s_{l+1} \cdots$ where $s_l \neq 0$, then apply the doubling rule l times. Remove the l bits from Seq_e.

We show a simple example. Assume that $e = 74539254$.

1. $Seq_e = 100100\bar{1}00\bar{1}0\bar{1}0\bar{1}000010000\bar{1}0\bar{1}0$
2. Set $\kappa = 4$.
3. $\langle 1, 2, 3, 5, 7, 9\rangle$.
4. Seq_e is parsed as follows.

$$\underbrace{1001}_{9}/00//\underbrace{\bar{1}001}_{-7}/0//\underbrace{\bar{1}0\bar{1}}_{-5}0/000//\underbrace{1}_{1}000/0//\underbrace{\bar{1}0\bar{1}}_{-5}0//,$$

where "/" means the parsing in Phase 1 and "//" stands for the parsing in Phase 2, i.e., the end of one cycle. The remaining AS-chain for e becomes $\langle 9, 18, 36//, 72, 144, 288, 576,_{(-7)} 569, 1138//, 2276, 4552, 9104,_{(-5)} 9099, 18198, 36396, 72792, 145584//, 291168,_{(+1)} 261169, 582338, 1164676, 2329352, 4658704//, 9317408, 18634816, 3726932,_{(-5)} 37269627, 74539254\rangle$,where ", " means the doubling rule while ",$_{(\pm\tilde{a})}$" stands for the star chain rule. Furthermore, "//," corresponds to the end of one cycle with the doubling rule. The total length of this chain is $32(= 5 + 27)$.

The average chain length l_{WS} in the window method for CSBR is given by the following theorem.

Theorem 2: The window method for CSBR satisfies[†]

$$\begin{aligned} l_{WS} \approx\ & d + \frac{q}{1-pq} - \left(\kappa - \frac{1}{(2-p_0)(1-p_0)}\right. \\ & \left. - \frac{(-(1-p_0))^\kappa}{2-p_0} + \frac{{p_0}^\kappa}{1-p_0}\right) \\ & + \frac{d + \frac{q}{1-pq}}{\kappa + \frac{1}{(2-p_0)(1-p_0)} + \frac{(-(1-p_0))^{\kappa-1}}{2-p_0}} \\ & + \frac{2^\kappa - (-1)^\kappa}{3}, \end{aligned} \tag{7}$$

where $p_0 = (1 - 3pq)/(1 - 2pq)$. Especially when $p = q = 1/2$, Eq. (7) becomes

$$\begin{aligned} l_{WS} \approx\ & \left(d + \frac{2}{3}\right) - \left(\kappa - \frac{4}{3} + \frac{1}{3}\frac{1}{4^{\lfloor\frac{\kappa-2}{2}\rfloor}}\right) \\ & + \frac{d + 2/3}{\kappa + \frac{4}{3} - \frac{1}{3}(-\frac{1}{2})^{\kappa-2}} + \frac{2^\kappa - (-1)^\kappa}{3}. \end{aligned} \tag{8}$$

Proof: The proof is given in [10]. □

l_{WB} and l_{WS} are shown for $d = 512$ and 1024 in Table 1.

We note that the window method can be constructed for any signed binary representation. Actually KT method [7] is a window method for a non-sparse signed binary representation, in which zero tends to run long. For any signed binary representation, the average length of the AS-chain is given by $d + d/A + B$, where A and B depend only on the window length and adopted signed binary representation. When $p = 1/2$, we note that from Eq. (8) the window method for the CSBR has $A \approx \kappa + \frac{4}{3}$ and $B = (2^\kappa - (-1)^\kappa)/3 - (\kappa - 2)$ for a little large κ, e.g., $\kappa \geq 5$. Now assume that the window method for a given (non-sparse) signed binary representation has $A = \kappa + \frac{4}{3} + \mu$ and $B = 2^{\kappa-1} - t - \kappa$ for some μ and t. KT method has $t = -3/4$ and $\mu = 1/6$ [7]. Then we have the following theorem.

Theorem 3: In case of the window method with $\kappa \geq 5$ for $p = 1/2$, the CSBR is more efficient than a given non-sparse signed binary representation if it satisfies

$$\mu \leqq 4 - \log(t + 13 - \mu). \tag{9}$$

Proof: Let κ_0 be the window length[††] that minimizes the chain length for the given non-sparse signed binary representation. Then letting $\kappa_c = \kappa_0 + \mu$ be the

[†] Although Eq. (7) does not hold with equality exactly, the right side is very tight approximation of l_{WS}.

[††] For simplicity, κ_0 is assumed to be real.

window length for the CSBR, which may not minimize the chain length of the CSBR, two representation have the same A. Hence, the CSBR is more efficient than the given signed binary representation if $(2^{\kappa_c}-(-1)^{\kappa_c})/3-(\kappa_c-2) < 2^{\kappa_0-1}-t-\kappa_0$. By substituting $\kappa_0 = \kappa_c - \mu$ and $\kappa_c = 5, 6, \cdots$ into this inequality, we obtain Eq. (9). □

Theorem 3 implies Corollary 1.

Corollary 1: The window method for the CSBR is more efficient than KT method.

Proof: By substituting $t = -\frac{3}{4}$ and $\mu = 1/6$ into Eq. (9), we obtain $1/6 \leqq 4 - \log(-\frac{3}{4}+13-\frac{1}{6}) \approx 0.405$. □

4. Extended Window Method

In the window method, the star chain rule in Eq. (5) is restricted to $\mathcal{D}_\kappa$, which consists of binary numbers with fixed length, or equivalently to $\tilde{\mathcal{D}}_\kappa$, which consists of all odd integer less than 2^κ. In the following, we remove this restriction, i.e., we assume that any set† of odd numbers $\tilde{\mathcal{D}}$ can be used instead of $\tilde{\mathcal{D}}_\kappa$. The extended window method can be roughly considered as a method which uses $\tilde{\mathcal{D}}$ instead of $\tilde{\mathcal{D}}_\kappa$, but it should be emphasized that many known methods, e.g. the window methods, MO method, KT method, etc., are special cases of the extended window method.

Roughly speaking, the A-chain of the extended window method is obtained by the following two steps.

Step 1. Construction of $\tilde{\mathcal{D}}$.

We first determine $\tilde{\mathcal{D}} = \{\tilde{a}_0(=1), \tilde{a}_1, \cdots, \tilde{a}_K\}$, each element of which is an odd number. Next we make an A-sequence for $\tilde{\mathcal{D}}$, i.e.,

$$\langle a_0(=\tilde{a}_0=1), a_1(=2), a_2, \cdots, a_{K+\delta}(=\tilde{a}_K)\rangle,$$
where $a_i = a_j + a_k$ $(j, k < i)$,

that includes all $\tilde{a}_j$, i.e. $\{a_0, \cdots, a_{K+\delta}\} \supseteqq \tilde{\mathcal{D}}$. Note that $\delta \geqq 1$ because $a_1 = 2$ must be included in any A-sequence, but the A-sequence for $\tilde{\mathcal{D}}$ should be devised in such a way that δ becomes as small as possible. Since K is not usually large and the difference between $\tilde{a}_i$ and $\tilde{a}_{i+1}$ is usually small, the optimal A-sequence for $\tilde{\mathcal{D}}$ can easily be obtained.

Step 2. Construction of A-chain for e

The remaining A-chain for e

$$\langle b_0(=\tilde{a}_k), b_1, b_2, \ldots, b_r(=e)\rangle, \text{ where } \tilde{a}_k \in \tilde{\mathcal{D}}$$

is obtained by applying the following two rules:

Doubling rule: $b_i = b_{i-1} + b_{i-1} = 2b_{i-1}$,
Star Chain rule: $b_i = b_{i-1} + \tilde{a}_j$, $\tilde{a}_j \in \tilde{\mathcal{D}}$.

By concatenating the above A-sequence and A-chain, we obtain the whole A-chain as follows.

$$\langle a_0(=1), a_1, \cdots, a_{K+\delta}, b_1, b_2, \cdots, b_r(=e)\rangle, \quad (10)$$

where the chain length is $r + K + \delta$.

We now describe the detailed, but general, algorithm to realize the above extended window method. The algorithm is described for the A-chain, but the case of AS-chain is shown in parentheses.

Detailed Algorithm of Extended Window Method

Step 0. Representation of e.

Represent a given positive integer e as the ordinary binary number (a given signed binary number), say Seq_e.

Step 1. Construction of $\tilde{\mathcal{D}}$.

(a) Determine $\tilde{\mathcal{D}} = \{\tilde{a}_0, \tilde{a}_1, \cdots, \tilde{a}_K\}$ in the following way. Determine binary sequences Seq_j, $j = 0, 1, 2, \cdots, K$ such that $\mathcal{D}_B = \{Seq_0, Seq_1 \cdots, Seq_K\}$ and 0^*, zero sequences with arbitrary length, can parse Seq_e uniquely. It is worth noting that since 0^* can be used (and $\overline{Seq_j}$ can be used in the case of of AS-chain), the MSB of every Seq_j should be one (Furthermore, if the CSBR is used for Seq_e, Seq_j should be sparse since Seq_e is sparse).

For each $Seq_j \in \mathcal{D}_B$, let Seq_j' be the binary (or signed binary) sequence obtained by removing less significant 0 running bits of Seq_j. Letting $\tilde{a}_j$ be the decimal number of Seq_j', we obtain $\tilde{\mathcal{D}} = \{\tilde{a}_0, \tilde{a}_1, \cdots, \tilde{a}_K\}$. Note that all $\tilde{a}_j$ are positive odd numbers.

(b) Make the shortest A-sequence (AS-sequence) $\langle a_0(=\tilde{a}_0=1), a_1, \cdots, a_K, \cdots, a_{K+\delta}\rangle$ that satisfies Eq. (2) ((3)) and $\{a_0, a_1, \cdots, a_{K+\delta}\} \supseteqq \tilde{\mathcal{D}}$.

Step 2. Construction of A-chain (AS-chain) for e.

Parse Seq_e from the MSB based on $\mathcal{D}_B = \{Seq_0, Seq_1, \cdots, Seq_K\}$ in the following way.

Phase 1 (a) Find Seq_j (Seq_j or $\overline{Seq_j}$), which equals to the prefix of Seq_e. Let $\tilde{a}_j$ be the decimal number of Seq_j' which is obtained from Seq_j.

(b) If Seq_j is the first parsing, then let $b_0 = \tilde{a}_j$, and go to step (d). Otherwise, apply the doubling rule $|Seq_j'|$ times.

(c) Apply the star chain rule: $b_i = b_{i-1} + \tilde{a}_j$ $\left(b_i = \begin{cases} b_{i-1} + \tilde{a}_j, & \text{if } Seq_j \text{ is used in (a)} \\ b_{i-1} - \tilde{a}_j, & \text{if } \overline{Seq_j} \text{ is used in (a)} \end{cases}\right)$

(d) Again apply the doubling rule $|Seq_j| -$

†Exactly, $\tilde{\mathcal{D}}$ must satisfy some conditions. See the detailed algorithm of the extended window method.

Table 2 Classification of previously proposed methods.

	BR/SBR	$\mathcal{D}_B$	$K+1$	512 bits
Binary [1]	BR	$\{1\}$	1	768
Window [1]	BR	all BS with length κ	$2^{\kappa-1}$	609.3 ($\kappa=5$)
MO [5]	CSBR	$\{1\}$	1	682
KT [7]	SBR(minimal non-sparse)	all SBS with length κ	$3^{\kappa-1}$	602.6 ($\kappa=5$)
EW with TA	BR	Adaptive (Algorithm 1)	—	609.3
	CSBR	Adaptive (Algorithm 2)		598.9
Window for CSBR	CSBR	all sparse SBS with length κ	$(2^{\kappa}-(-1)^{\kappa})/3$	598.9 ($\kappa=6$)

BR/BS: Ordinary binary representation /sequence.
SBR/SBS: Signed binary representation /sequence.
EW with TA: Extended window method with Tunstall-like Algorithm
The MSB of BS and SBS in $\mathcal{D}_B$ is 1. κ is an integer.

$|Seq_j'|$ times.

(e) Remove the prefix Seq_j (Seq_j or $\overline{Seq_j}$) from Seq_e.

Phase 2 (f) If the prefix of Seq_e is $\underbrace{0\cdots0}_{t}1$ ($\underbrace{0\cdots0}_{t}1$ or $\underbrace{0\cdots0}_{t}\bar{1}$), then apply the doubling rule t-times.

(g) Remove the prefix $\underbrace{0\cdots0}_{t}$ from Seq_e.

If Seq_e becomes null sequence in Phase 1 or 2, then finish this algorithm. Otherwise, repeat Phase 1 and Phase 2.

We call "Phase 1 + Phase 2" one cycle. Note that in Phase 1 of Step 2 any Seq_j happen not to be a prefix of Seq_e when the length of Seq_e becomes short. In such a case, find Seq_j that satisfies Seq_j (Seq_j or $\overline{Seq_j}$) $= Seq_e\underbrace{00\cdots0}_{t}$, and perform (b) and (c). Finally, apply the doubling rule $|Seq_j| - |Seq_j'| - t$ times in (d).

How to construct optimal $\mathcal{D}_B = \{Seq_j\}$ and the computation complexity of this process will be treated in Sect. 7. Note that this complexity need not be included in the computational complexity of M^e (or eM) because e is fixed in our case and $\mathcal{D}_B$ depends only on e. In order to parse Seq_e based on $\mathcal{D}_B$ and 0^* in the extended window method, we need a little additional task compared with the window method. However the computation complexity of the parsing is $O(\log K)$ by using the so–called parsing tree. Since $K \ll d$ holds, the time of parsing is negligible.

We show an example of the CSBR for $e = 74539254$, which is the same treated in Sect. 3.

Step 0. Representation of e.

$Seq_e = 100100\bar{1}00\bar{1}0\bar{1}0\bar{1}0000\bar{1}0000\bar{1}0\bar{1}0$

Step 1. Construction of $\tilde{\mathcal{D}}$

(a) $\mathcal{D}_B = \{Seq\} = \{1000, 1010, 10\bar{1}0, 10010, 100\bar{1}0\}$ with $K+1 = 5$, $\tilde{\mathcal{D}}_B = \{Seq'\} = \{1, 101, 10\bar{1}, 1001, 100\bar{1}\}$, $\tilde{\mathcal{D}} = \{\tilde{a}\} = \{1, 5, 3, 9, 7\}$.

(b) $\langle 1, 2, 3, 5, 7, 9\rangle$.

Step 2. Construction of the AS-chain for e

Seq_e is parsed as follows.

$$\underbrace{1001}_{9}0/0//\underbrace{\bar{1}001}_{-7}0//\underbrace{\bar{1}0\bar{1}}_{-5}0/000//\underbrace{1}_{1}000/0//\underbrace{\bar{1}0\bar{1}}_{-5}0//,$$

where "/" means the parsing in Phase 1 and "//" stands for the parsing in Phase 2, i.e., the end of one cycle. Although the parsing is slightly different from the one described in Sect. 3, the exactly same AS-chain is obtained.

Table 2 classifies known methods in terms of the extended window method, i.e., based on the representation of Seq_e, $\mathcal{D}_B$, and the size $K+1$ of $\tilde{\mathcal{D}}$. In the table, the window method for CSBR and the extended window method with Tunstall-like algorithm described in Sect. 7 is included. The last column shows the average chain length obtained by computer simulation when e is randomly generated with 512 bits. Note that any methods included in the extended window method can easily be implemented.

Yacobi [6] and Bos-Coster [4] proposed other methods for the A-chain, which cannot be classified in the extended window method. Yacobi's method is based on LZ78 code, which is one of universal data compression codes, and the size of $\mathcal{D}_B$ is not fixed. We note from the computer simulation that the average chain length attained by his method is about 635 when e is 512 bits, and hence it is inefficient. Bos-Coster's method

is a heuristic method with a large window, and the average chain length attained by their method is about 605 when e is 512 bits.

5. Lower Bound of Average Chain Length

5.1 Chain Length Attained by the Extended Window Method

We now analyze the chain length attained by the extended window method for A-chain and AS-chain with CSBR. We assume that e is randomly generated, but $d = |Seq_e| = \lfloor \log e \rfloor + 1$ is fixed, and each bit of Seq_e except the MSB is 0 with probability p in the ordinary binary representation or with p_0 in the CSBR. The length r shown in Eq. (10) is explained as r =(the number of applied star chain rule) + (the number of applied doubling rule). Since, in steps (b), (d) and (f), the doubling rule is applied to every bit of Seq_e except the first parsing $Seq_j'(1)$, the latter number equals to $d - |Seq_j'(1)|$. Similarly, since the star chain rule is applied for every cycle as shown in (c) of Phase 1, its average number is given by d/L_{cycle}, where L_{cycle} is the average length of one cycle. Hence, the average length of total chain l_E is given by

$$l_E = d - |Seq_j'(1)|_{av} + \frac{d}{L_{cycle}} + K + \delta, \tag{11}$$

where $|Seq_j'(1)|_{av}$ is the average length of $Seq_j'(1)$. Letting P_j be the probability of Seq_j or $\overline{Seq_j}$ in (a) of Phase 1, the second and third terms of Eq. (11) can be represented as follows.

$$|Seq_j'(1)|_{av} = \sum_{j=0}^{K} |Seq_j'|P_j \leqq \sum_{j=0}^{K} |Seq_j|P_j, \tag{12}$$

$$\begin{aligned} L_{cycle} &\leqq \sum_{l=0}^{\infty}\sum_{j=0}^{K}(|Seq_j| + l)P_j \cdot q_j(l) \\ &= \begin{cases} \sum_{j=0}^{K} |Seq_j|P_j + p/(1-p)\ , \\ \quad \text{in case of A-Chain} \\ \sum_{j=0}^{K} |Seq_j|P_j + p_0/(1-p_0)\ , \\ \quad \text{in case of AS-Chain for CSBR} \end{cases} \end{aligned} \tag{13}$$

where $q_j(l)$ is the probability that l continuous 0's occur after Seq_j, and it is given by $q_j(l) = p^l(1-p)$ in the case of A-chain and $q_j(l) = {p_0}^l(1-p_0)$ in the case of AS-chain with CSBR. The first term of Eq. (13) equals to the average length of Phase 1 while the second term is a bound of the average length of Phase 2.

5.2 Lower Bound of Chain Length

In this section, we derive lower bounds of l_E in the extended window method for given d and p (or p_0). First we consider the case of A-chain. Since $|Seq_j'(1)|_{av}$ in Eq. (11) becomes negligible compared with the last two terms as d becomes bigger, the last two terms should be minimized to attain the shortest A-chain for large d. Hence, from Eqs. (11) and (13), we minimize the following function.

$$f(\mathcal{D}_B) = \frac{d}{\sum_{j=0}^{K} |Seq_j|P_j + \frac{p}{1-p}} + K + \delta. \tag{14}$$

We separate this minimization problem into two sub-problems.

Problem A: Obtaining the optimal $\mathcal{D}_B = \{Seq_j\}$ which maximize $\sum_{j=0}^{K} |Seq_j|P_j$ for a given K. Let $g(K)$ be the maximal value.

Problem B: Determine the optimal K that minimizes $\frac{d}{g(K)+\frac{p}{1-p}} + K + \delta$.

Concerning these problems, the following lemmas hold.

Lemma 1: $\sum_{j=0}^{K} |Seq_j|P_j$ satisfies

$$\sum_{j=0}^{K} |Seq_j|P_j \leqq 1 + \frac{\log(K+1)}{H(p)}\ , \tag{15}$$

where $H(p)$ is the binary entropy function given by $H(p) \equiv -p\log p - (1-p)\log(1-p)$.

Proof: Problem A is equivalent to the optimization problem of VF code shown in the Appendix. Hence, from Eq. (A· 1) in the Appendix, we obtain the following inequality.

$$\begin{aligned} \sum_{j=0}^{K} |Seq_j|P_j &= \sum_{j=0}^{K}(|Seq_j| - 1)P_j + 1 \\ &\leqq \frac{\log(K+1)}{H(p)} + 1, \end{aligned} \tag{16}$$

where one is subtracted in $\sum_{j=0}^{K}(|Seq_j| - 1)P_j$ because the MSB of Seq_j is always "1." □

Lemma 2: $f(\mathcal{D}_B)$ has the following lower bound.

$$f(\mathcal{D}_B) \geqq \frac{H(p)d}{\log(K+1) + \frac{1}{1-p}H(p)} + K + \delta\ , \tag{17}$$

where $H(p)d$ is the minimum attainable length when Seq_e is compressed in source coding.

Proof: By substituting Eq. (15) into Eq. (14), we obtain

$$\begin{aligned} f(\mathcal{D}_B) &\geqq \frac{d}{\frac{\log(K+1)}{H(p)} + 1 + \frac{p}{1-p}} + K + \delta \\ &= \frac{H(p)d}{\log(K+1) + \frac{1}{1-p}H(p)} + K + \delta. \end{aligned}$$

□

Letting the right side of Eq. (17) be $G(K)$, Problem B is equivalent to the problem of determining the

optimal K that minimizes $G(K)$. In small K, $G(K)$ decreases because the first term dominates it while in large K, $G(K)$ increases because of the second term. This means that the optimal K exists.

Noting that $|Seq'_j(1)|_{av} = \sum_{j=0}^{K} |Seq'_j| P_j \leqq \sum_{j=0}^{K} |Seq_j| P_j$, we have the following theorem from Lemmas 1 and 2.

Theorem 4: The A-chain length of the extended window method, l_{EB}, satisfies

$$l_{EB} \geqq d - \left(\frac{\log(K+1)}{H(p)} + 1\right) + \frac{H(p)d}{\log(K+1) + \frac{1}{1-p}H(p)} + K + \delta. \quad (18)$$

Equation (18) holds only for the A-chain obtained by the extended window method. For general addition rule Eq. (2), Schönhage [13] showed that the chain length l must satisfy $l \geqq d + \log(1-p)d - 2.13$ for A-chain of all e.

Next, we consider the case of AS-chain. Generally it is difficult to evaluate l_E for any AS-chain because each bit of signed binary sequence Seq_e cannot be treated as memoryless even when e is randomly generated. However, in the case of CSBR, each bit can be considered as an output of a first-order Markov source and its sequence can be transformed into a memoryless sequence by removing every "0" just after a nonzero bit. This *reduction transformation* is possible because of the sparse property of the CSBR. We also note from Appendix A.2 that the average length of Seq_e is given by $d + q/(1-pq)$ when Seq_e is the CSBR. Furthermore, the transformed sequence of Seq_e becomes a memoryless sequence with average length

$$d_t = \left(d + \frac{q}{1-pq}\right)\frac{1-2pq}{1-pq},$$

and the probability of symbol $j \in \{\bar{1}, 0, 1\}$, p_j, in the transformed sequence is given by $p_0 = \frac{1-3pq}{1-2pq}$, $p_1 = \frac{p^2 q}{1-2pq}$, $p_{\bar{1}} = \frac{pq^2}{1-2pq}$. Letting S_j $(0 \leqq j \leqq K)$ be the transformed sequence obtained from Seq_j, $f(\mathcal{D}_B)$ is given by

$$f(\mathcal{D}_B) = \frac{d_t}{\sum_{j=0}^{K} |S_j| P_j + \frac{p_0}{1-p_0}} + K + \delta. \quad (19)$$

In the same way as the A-chain case, we obtain

$$\sum_{j=0}^{K} |S_j| P_j \leqq 1 + \frac{\log(K+1)}{H(X)}, \quad (20)$$

where $H(X) = -\sum_{i=-1}^{1} p_i \log p_i$ is the entropy of the transformed sequence. By substituting Eq. (20) into Eq. (19), $f(\mathcal{D}_B)$ is bounded by

$$f(\mathcal{D}_B) \geqq \frac{H(X)d_t}{\log(K+1) + \frac{1}{1-p_0}H(X)} + K + \delta. \quad (21)$$

Note that since $H(X)d_t$ is the minimum binary length attained by compressing the transformed sequence of Seq_e, it is bounded by $H(p)d$ because $H(p)d$ is the lower bound of any binary representation of e. Since $H(p) = 1, H(X) = 3/2$, and $p_0 = 1/2$ in the case of $p = 1/2$, we note by comparing Eqs. (17) with (21) that the AS-chain possibly becomes shorter than the A-chain.

Letting $S'_j(1)$ be the transformed sequence of $Seq'_j(1)$, we have, in the same way as the A-chain, that the average length of $S'_j(1)$ is bounded by $\log(K+1)/H(X)+1$. Noting that $|Seq'_j(1)| \leqq 2|S'_j(1)|$, we have the following theorem.

Theorem 5: In the extended window method, the AS-chain length for the CSBR, l_{ES}, satisfies

$$l_{ES} \geqq \left(d + \frac{q}{1-pq}\right) - 2\left(\frac{\log(K+1)}{H(X)} + 1\right) + \frac{H(X)d_t}{\log(K+1) + \frac{1}{1-p_0}H(X)} + K + \delta \quad (22)$$

Table 1 shows the comparison between l_{WB} and l_{WS} given by Eq. (6) or (8), where the optimal κ is searched by computer simulation, and the lower bounds of l_{EB} and l_{ES} given by Eq. (18) or (22), where the optimal $K+1$ is found by exhaustive search. From the table, we note that the window methods are almost optimal for $p = 1/2$ since their performance is very close to the lower bound.

6. Asymptotic Performance of Extended Window Methods

In this section, we consider the asymptotic performance of the extended window methods for the ordinary binary representation and the CSBR.

In inequalities (13) and (15), the difference between the right and left sides goes to zero when $d \to \infty, K \to \infty$, and $K/d \to 0$. Hence, for such large d and K, we have

$$l_{EB} \approx d - \left(\frac{\log(K+1)}{H(p)} + 1\right) + \frac{H(p)d}{\log(K+1) + \frac{1}{1-p}H(p)} + K + \delta. \quad (23)$$

By substituting $\log(K+1) = \log d - 2\log^2 d$, we obtain

$$l_{EB} \leqq d + H(p)\frac{d}{\log d} + \mathrm{O}\left(\frac{d\log^2 d}{(\log d)^2}\right). \quad (24)$$

Similarly, the following inequality holds for l_{ES}.

$$l_{ES} \leqq d + H(X)\frac{1-2pq}{1-pq}\frac{d}{\log d} + \mathrm{O}\left(\frac{d\log^2 d}{(\log d)^2}\right). \quad (25)$$

On the other hand, it is well known [1] that for sufficiently large d, l_{WB} satisfies

$$l_{WB} \leqq d + \frac{d}{\log d} + \mathrm{O}\left(\frac{d \log^2 d}{(\log d)^2}\right). \tag{26}$$

Similarly l_{WS} also satisfies that $l_{WS}(d) \leqq d + \frac{d}{\log d} + \mathrm{O}\left(\frac{d \log^2 d}{(\log d)^2}\right)$.

Comparing Eqs. (24) and (26) and noting that $0 \leqq H(p), H(X)\frac{1-2pq}{1-pq} \leqq 1$, we may say that for sufficient large d the extended window method is more efficient than the window method except the case of $p \neq 1/2$[†]. The same is also true when $H(X)\frac{1-2pq}{1-pq} \neq 1$, i.e. $p \neq 1/2$, for CSBR.

When $p = 1/2$, we have that $H(p) = H(X)\frac{1-2pq}{1-pq} = 1$. Hence the bounds Eqs. (24) and (25) coincide with the well known results that the length of A-chain with general rule Eq. (2) can attain $d + d/\log d$ asymptotically [14]. But when $p \neq 1/2$, Eqs. (24) and (25) means that we can attain shorter chain length than $d + d/\log d$ asymptotically.

7. Optimal Algorithm to Construct $\mathcal{D}_B$

We now show how to construct the optimal $\mathcal{D}_B = \{Seq_j | 0 \leqq j \leqq K\}$ for a given size $K+1$ in the extended window method. As noted in Sect. 5, the optimization problems A and B[††] are equivalent to the optimization problem of VF code. Hence, the optimal sequences $\mathcal{D}_B = \{Seq_j\}$ can be obtained by using Tunstall algorithm [11], [12], which derive the optimal VF code, as follows. The first one is for the A-chain while the second one is for the AS-chain of CSBR.

Algorithm 1 (Tunstall-like Algorithm for A-chain)

1. Make the root of a tree with weight 1.
2. While the number of leaves is less than $K+1$, repeat the following.

   Let l and $weight(l)$ be the leaf with largest weight and its weight, respectively. Create two children with weight $p \times weight(l)$ and weight $q \times weight(l)$, and connect them to l via edges labeled with "0" and "1," respectively.
3. Get binary sequences by reading along the edges from the root to leaves, and construct $\mathcal{D}_B = \{Seq_j | 0 \leqq j \leqq K\}$ by concatenating "1" to each sequence as the MSB.

In order to accomplish this algorithm, we must compare two leaf weights in Step2, totally $K(K+1)/2$ times to find the leaf with largest weight and we also need some additional tasks to construct the tree. But the computational complexity is small and the algorithm can easily be implemented.

In the case of AS-chain for CSBR, canonical signed binary sequences can be transformed into memoryless sequences with probability $p_j, j \in \{\bar{1}, 0, 1\}$, by the reduction transformation described in Sect. 5. Since we can use both Seq_j and $\overline{Seq_j}$ in the parsing, we construct $\mathcal{D}_B$ based on probability $\hat{q} \equiv (p_1 + p_{\bar{1}})/2 = (1 - p_0)/2$.

Algorithm 2 (Tunstall-like Algorithm for AS-chain with CSBR)

1. Make the root of a tree with weight 1.
2. While the number of leaves is less than $K+1$, repeat the following.

   Let l and $weight(l)$ be the leaf with largest weight and its weight, respectively. Create three children with weight $p_0 \times weight(l)$, weight $\hat{q} \times weight(l)$ and $\hat{q} \times weight(l)$, and connect them to l via edges labeled with "0", "10" and "$\bar{1}$0," respectively.
3. Get signed binary sequences by reading along the edges from the root to leaves, and construct $\mathcal{D}_B = \{Seq_j | 0 \leqq j \leqq K\}$ by concatenating "10" to each sequence as the MSBs.

The computation complexity of Algorithm 2 is almost same as Algorithm 1. It is worth noting that the extended window method with Tunstall-like algorithms (Algorithms 1 and 2) asymptotically attain the bound of Eqs. (24) and (25), respectively.

Next we consider the problem to optimize the size of $\mathcal{D}_B$, $K+1$. However, since it is difficult to derive the optimal K analytically, we must find it by exhaustive search. When e is a sufficiently large random number, we can assume by the law of large number that $p = 1/2$. Hence, in the followings, we treat the case of $p = 1/2$, which is important in practice.

First we consider the case of A-chain. We can show by exhaustively search that for almost all d, $f(\mathcal{D}_B)$ of Eq. (14) is minimized when $K+1 = 2^{\kappa-1}$ for some positive integer κ. For instance, the optimal K's are given by $K = 15 = 2^4 - 1$ for $d = 512$, $K = 31 = 2^5 - 1$ for $d = 1024$, $K = 63 = 2^6 - 1$ for $d = 2048$, etc. When $K+1$ equals to $2^{\kappa-1}$, all sequences $Seq_j \in \mathcal{D}_B$ obtained by Algorithm 1 have the same length κ. Since such sequences are equal to the sequences used in the usual window method, we can conclude that in the case of A-chain, the usual window method is almost *optimal* when d is sufficiently large.

In the case of AS-chain with CSBR, we can also show by exhaustive search that for almost all d, $f(\mathcal{D}_B)$ of Eq. (19) is minimized when $K+1$ equals to $(2^\kappa - (-1)^\kappa)/3$. For example, $K = 20 = (2^6 - (-1)^6)/3 - 1$ for $d = 512$ and $d = 1024$, $K = 42 = (2^7 - (-1)^7)/3 - 1$ for $d = 2048$, etc. When $K+1$ equals to $(2^\kappa - (-1)^\kappa)/3$, each sequence Seq_j obtained by Algorithm 2 satisfies the following properties. 1) The MSB is "1." 2) The

[†] In fact we prove in Sect. 7 that the window method is almost optimal in the case of $p = 1/2$ for any large d.

[††] Since $f(\mathcal{D}_B)$ is minimized instead of l_{EB} (or l_{ES}), the derived $\mathcal{D}_B$ may be sub-optimal. But, it becomes optimal when d is sufficiently large.

length of sequence is κ or $\kappa+1$. 3) The least significant bit (LSB) is always "0." 4) When the length is $\kappa+1$, the second LSB is nonzero. We note from these properties that when the above Seq_j has length $\kappa+1$, we can remove the $(\kappa+1)$-th bit of Seq_j without increasing the AS-chain length by postponing the process of $(\kappa+1)$-th 0 bit to the second phase[†]. Hence, by such truncation, all Seq_j's have the same length κ. This means that for CSBR, the window method is almost *optimal* among the extended window method.

We now consider the reason why the A-chain is apt to be optimal at $K+1=2^{\kappa-1}$. For $2^{\kappa-2}<K+1\leqq 2^{\kappa-1}$, each Seq_j occurs with probability $2^{-(\kappa-2)}$ or $2^{-(\kappa-1)}$. Hence, a gap of probability occurs at $K+1=2^{\kappa-1}$. Furthermore, δ becomes the minimum, i.e. one, at $K+1=2^{\kappa-1}$. Therefore, the average chain length is apt to be minimized at $K+1=2^{\kappa-1}$. The similar argument holds for the AS-chain for CSBR.

Since there is little space to discuss the case of $p\neq 1/2$, we show only an example that the extended window method is more efficient than the window method. The optimal sequences $\{Seq_j\}$ for e with $d=512$ and $p=0.15$ is given by $\{100, 1010, 1100, 10110, 10111, 11010, 11011, 11101, 11100, \underbrace{11\cdots1}_{4,\cdots,14}0, \underbrace{1\cdots1}_{15}\}$, and the chain length is only 603 while the chain length of window method ($\kappa=6$) is 621. Some results for the case of $p\neq 1/2$ are shown in [15]. But the detailed study for such case will be reported in another paper.

8. Conclusion

In this paper, the window method and the extended window method to calculate M^e or eM were proposed, and the theoretical bounds of the shortest AS-chain length were derived. Furthermore, we showed that the optimal $\mathcal{D}_B=\{Seq_j\}$ used in the extended window method can be obtained by Tunstall-like Algorithm, and the extended window method with optimal $\mathcal{D}_B$ can attain the bound asymptotically. We also showed that even for finite e, the extended window method is more efficient than other known methods.

References

[1] D.E. Knuth, "Seminumerical algorithm (arithmetic) The art of Computer Programming vol.2," Addison Wesley, 1981.
[2] Y. Tsuruoka and K. Koyama, "Fast exponentiation algorithm based on batch-processing and precomputation," IEICE Trans. Fundamentals, vol.E80-A, no.1, pp.34–39, Jan. 1997
[3] P. Downey, B. Leong, and R. Sthi, " Computing sequences with addition chains," SIAM J. Computing, vol.10, no.3, pp.638–646, 1981.
[4] J. Bos and M. Coster, "Addition chain heuristics," Proc. of CRYPTO'89, pp.400–407, 1989.
[5] F. Morain and J. Olivos, "Speeding up the computations on an elliptic curve using addition–subtraction chains," Theoretical Informatics and Applications, vol.24, no.6, pp.531–544, 1990.
[6] Y. Yacobi, "Exponentiating faster with addition chains," Proc. of EUROCRYPTO '90, pp.222–229, 1990.
[7] K. Koyama and Y. Tsuruoka, "Speeding up elliptic cryptosystems by using a signed binary window method," Proc. of CRYPTO'92, pp.345–357, 1992.
[8] K. Hwang, "Computer Arithmetic," Wiley, New York, 1979.
[9] G.W. Reitweisner, "Binary Arithmetic (Advances in Computers, vol.1)," pp.232–308, Academic Press, New York, 1960.
[10] N. Kunihiro, "Study on fast algorithms for exponentiation," M.E. Thesis, Dept. of Math. Eng. and Inform. Physics, Univ. of Tokyo, 1996.
[11] B.P. Tunstall, "Synthesis of noiseless compression codes," Ph.D. thesis, Georgia Inst. Tech., Atlanta, 1968.
[12] P.R. Stubley, "Adaptive data compression using tree codes," Ph.D. thesis, University of Waterloo, Ontario, 1992.
[13] A. Schönhage, "A lower bound for the length of addition chains," Theoretical Computer Science, vol.1, pp.1–12, 1975.
[14] P. Erdös, "Remarks on Number Theory III, on Addition Chains," Acta Arith. VI, pp.77–81, 1960.
[15] N. Kunihiro and H. Yamamoto, "Optimal addition chain classified by Hamming weight," IEICE Technical Report, ISEC96-74, March 1997.

Appendix

A.1: Optimal VF Source Code

The optimization problem of VF (Variable to Fixed) source code is studied in [11], [12], etc. In VF coding, a source sequence is parsed based on a set of sequences $\{Seq_1, Seq_2, \cdots, Seq_K\}$, where each length of Seq_i is not fixed, but $\{Seq_1, Seq_2, \cdots, Seq_K\}$ is usually required to satisfy the prefix condition for instantaneous encoding. Each Seq_i is encoded to a codeword with $\lceil \log K \rceil$ bits.

Letting t_i and P_i be the length of Seq_i and its probability, respectively, the average codeword length per one source symbol is given by $\overline{L}=\log K/\sum_{i=1}^{K} t_i P_i$. In order to minimize $\overline{L}$ we need to optimize codeword size K and codewords $\mathcal{D}_B=\{Seq_i\}$.

The above problem can be separated into the following two optimization problems.

$$\text{Problem A}: g(K)\equiv \max_{\mathcal{D}_B}\sum_{i=1}^{K} t_i P_i,$$

$$\text{Problem B}: \min_{K}\frac{\log K}{g(K)}$$

Since it is well-known from the source coding theorem that $\overline{L}$ must be greater than the source entropy $H(X)$, the following inequality must hold.

[†]See the example in Sects. 3 and 4. In fact, $\mathcal{D}_B$ shown in Sect. 4 is constructed by Algorithm 2 under the condition $p_0=1/2$ and $K+1=5$.

$$\min_K \frac{\log K}{g(K)} \geq H(X) \qquad \text{(A·1)}$$

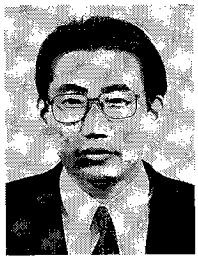
When the sequence is memoryless, we know the solutions of these two problems. For Problem A, the optimal $\mathcal{D}_B$ to maximize $\sum_{i=1}^{K} t_i P_i$ is obtained by Tunstall algorithm [11]. Concerning Problem B, it is shown that when we use Tunstall algorithm, the left side of Eq. (A·1) asymptotically converges to $H(X)$ as K increases.

A.2: Canonical Signed Binary Representation

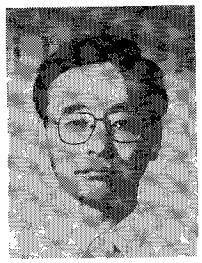
Assume that by Booth algorithm, Seq_e is derived from d bits ordinary binary sequence $Seq_e^{(d)}$, each bit of which, except the MSB, is a memoryless source output with $p = \Pr\{s = 0\}$ and $q = 1 - p = \Pr\{s = 1\}$. In Booth algorithm, one bit is inserted only if $Seq_e^{(d)} = 1(01)^\ell 1*$, $\ell = 0, 1, \cdots$, where $*$ and $(01)^\ell$ stands for arbitrary sequence and ℓ repetition of "01", respectively. Since the MSB is always one, such case occurs with probability

$$\sum_{\ell=0}^{\lfloor \frac{d-2}{2} \rfloor} (qp)^\ell q = \frac{q(1 - (qp)^{\lfloor \frac{d-2}{2} \rfloor + 1})}{1 - pq},$$

which can be approximated very closely by $q/(1-pq)$ if $d \geq 20$. Note that we usually treat much larger d. Hence, the average length of Seq_e becomes $d + q/(1 - pq)$ bits. Let p_j be the probability such that symbol "j" occurs after symbol "0". Then, it can be easily shown that $p_0 = \frac{1-3pq}{1-2pq}, p_1 = \frac{p^2 q}{1-2pq}$, and $p_{\bar{1}} = \frac{pq^2}{1-2pq}$. For $p = 1/2$, we have $p_0 = 1/2$ and $p_{\bar{1}} = p_1 = 1/4$. Furthermore, letting S_e be the transformed sequence of Seq_e described in Sect. 3, we obtain from the relation $|Seq_e| = |S_e|(p_0 + 2p_1 + 2p_{\bar{1}})$ that $|S_e| = |Seq_e|(1-2pq)/(1-pq)$.



Noboru Kunihiro was born in Aichi, Japan on June 26, 1971. He received the B.E. and M.E. in mathematical engineering and information physics from University of Tokyo, Tokyo, Japan in 1994 and 1996, respectively. Since 1996 he has been with NTT Communication Science Laboratories and engaged in research for cryptography and information security. His research interest includes elliptic curve theory and cryptography. He was awarded the SCIS'97 paper prize.



Hirosuke Yamamoto was born in Wakayama, Japan, on November 15, 1952. He received the B.E. degree from Shizuoka University, Shizuoka, Japan, in 1975 and the M.E. and Dr.E. degrees from the University of Tokyo, Tokyo, Japan, in 1977 and 1980, respectively, all in electrical engineering. In 1980 he joined Tokushima University, Tokushima, Japan. He was an Associate Professor at Tokushima University from 1983 to 1987, and at University of Electro-Communications, Tokyo, Japan, from 1987 to 1993. Since 1993 he has been an Associate Professor in the Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokyo, Tokyo, Japan. In 1989–90, he was a Visiting Scholar at the Information Systems Laboratory, Stanford University. His research interests are in Shannon theory, coding theory, cryptology, and communication theory. Dr. Yamamoto is a member of the IEEE.