PAPER Special Section on Information Theory and Its Applications

A Heuristic Search Method with the Reduced List of Test Error Patterns for Maximum Likelihood Decoding*

Hideki YAGI[†,††a)], ***Student Member***, **Toshiyasu MATSUSHIMA**[†], ***Member***, ***and*** **Shigeichi HIRASAWA**[†], ***Fellow***

SUMMARY The reliability-based heuristic search methods for maximum likelihood decoding (MLD) generate test error patterns (or, equivalently, candidate codewords) according to their heuristic values. Test error patterns are stored in lists and its space complexity is crucially large for MLD of long block codes. Based on the decoding algorithms both of Battail and Fang and of its generalized version suggested by Valembois and Fossorier, we propose a new method for reducing the space complexity of the heuristic search methods for MLD including the well-known decoding algorithm of Han et al. If the heuristic function satisfies a certain condition, the proposed method guarantees to reduce the space complexity of both the Battail-Fang and Han et al. decoding algorithms. Simulation results show the high efficiency of the proposed method.

key words: *maximum likelihood decoding, binary block codes, heuristic search, most reliable basis, reliability*

1. Introduction

Maximum likelihood decoding (MLD) of block codes minimizes the probability of decoding error when we assume that each codeword has the equal probability to be transmitted. Since the complexity of searching the most likely codeword is significantly large, many researchers have devoted to develop efficient algorithms for MLD of long block codes. One of the most efficient MLD algorithms is the reliability-based decoding algorithm that uses the column permuted generator matrix in non-increasing order of reliability.

In general, the reliability-based decoding algorithms are divided into two types due to the generation rule of candidate codewords. The first type of them generates the candidate codewords according to a predetermined generation rule [4], [5], [10]. The latter one is called the *heuristic search MLD algorithms* where candidate codewords are generated in increasing value of the heuristic function (also called the evaluation function) [1]–[3], [6]–[9]. Test error patterns (information sequences corresponding to candidate codewords) are generated and stored in lists before they are tested to be the most likely codeword. In this paper, we will consider the latter one. As known to the authors, G. Battail and J. Fang first proposed a heuristic search method for MLD over the additive white Gaussian noise (AWGN) channel [1] (we will call this method the BF decoding algorithm). Recently, in [8], [9], A. Valembois and M. Fossorier have indicated that a generalized version of the BF decoding algorithm is equivalent to the well-known A* decoding algorithm proposed by Y.S. Han et al. [2]. The generalized BF (GBF) decoding algorithm is a prominent and effective algorithm which can deal with almost all heuristic functions ever proposed.

For heuristic search MLD algorithms, their memory management is the critical issue since the maximum list size of test error patterns (TEPs), which dominates the space complexity, becomes quite large as the signal to noise ratio (SNR) of the channel decreases. There are roughly three approaches to reduce the maximum list size of TEPs in heuristic search MLD algorithms: (i) Some studies have proposed effective heuristic functions of TEPs to early terminate decoding procedure before the list of TEPs becomes very large [6], [7]. (ii) Some studies have proposed techniques for reducing the maximum list size of TEPs employing conventional heuristic functions [8]. (iii) Some studies have discarded the optimality of decoding while the complexity of decoding is drastically reduced [3].

Valembois et al. have taken the second approach. They have proposed a technique which considerably reduces the maximum list size of TEPs of the original BF decoding algorithm which imposes some condition for heuristic functions [8]. However, their technique cannot be adopted to the GBF decoding algorithm in which the search is guided by more effective heuristic functions than that considered in [1].

In this paper, we also consider the second approach and propose a method for reducing the maximum list size of TEPs of the GBF decoding algorithm. Similarly to the Valembois' approach, we first define a condition of heuristic functions. We show that the defined condition is satisfied by most of well-known heuristic functions. Then, we propose the improved method for the GBF decoding algorithm when the heuristic function satisfies the defined condition. We also devise the adaptive procedure of the proposed method where the heuristic function is updated as decoding proceeds. Proposed methods guarantee to reduce the maximum list size of TEPs of the GBF decoding algorithm. The number of TEPs generated and stored in lists are reduced and so they also reduce the time complexity of the GBF decoding algorithm. We also show by computer simulations that the space complexity of the GBF decoding (or equiva-

Manuscript received January 24, 2005.
Manuscript revised April 22, 2005.
Final manuscript received June 13, 2005.

[†]The authors are with the Department of Industrial and Management Systems Engineering, Waseda University, Tokyo, 169-8555 Japan.
[††]The author is with Media Network Center, Waseda University, Tokyo, 169-0051 Japan.

*The content of this paper is partly based on [11], [12].

a) E-mail: yagi@hirasa.mgmt.waseda.ac.jp

DOI: 10.1093/ietfec/e88–a.10.2721

lently, the A* decoding) algorithm is significantly reduced.

This paper is organized as follows. In Sect. 2, we briefly review general reliability-based MLD algorithms. In Sect. 3, we describe some heuristic functions and the GBF decoding algorithm. In Sect. 4 and 5, we propose new methods for reducing the space complexity of the GBF decoding algorithms. In Sect. 6, we show some simulation results and finally we state the concluding remarks in Sect. 7.

2. Reliability-Based MLD Algorithm

Let C be a binary linear (n, k, d) block code of the code length n, the number of information symbols k and the minimum distance d. We denote a generator matrix of C by G and the weight profile of C by $W(C)$. We assume any codewords $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \{0,1\}^n$ of C are transmitted over the AWGN channel. The receiver maps the received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$ into the reliability sequence $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$, $\theta_j = \ln \frac{P(r_j|c_j=0)}{P(r_j|c_j=1)}$, where $P(r_j|c_j)$ represents the likelihood of the symbol[†] c_j. Furthermore, the hard-decision received sequence $\boldsymbol{z} = (z_1, z_2, \ldots, z_n) \in \{0,1\}^n$ is obtained by setting $z_j = 0$ if $\theta_j \geq 0$ and $z_j = 1$ otherwise. The decoder estimates the transmitted codeword both from $\boldsymbol{\theta}$ and $\boldsymbol{z}$.

In reliability-based decoding algorithms, we permute columns of a generator matrix in non-increasing order of reliability so that the leftmost k positions are the *most reliable and linearly independent* (MRI) [2], [6], [8], [9]. The other columns outside the k MRI positions are also reordered in non-increasing order of reliability, i.e., $|\theta_{j_1}| \geq |\theta_{j_2}|$ for $1 \leq j_1 < j_2 \leq k$ and for $k+1 \leq j_1 < j_2 \leq n$. We perform the standard row operations with respect to the permuted matrix to make the leftmost k columns the identity matrix. We denote the resultant matrix by $\tilde{G}$.

Let $\tilde{\boldsymbol{\theta}} = (\tilde{\theta}_1, \tilde{\theta}_2, \ldots, \tilde{\theta}_n)$ and $\tilde{\boldsymbol{z}} = (\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n)$ be permuted sequences of $\boldsymbol{\theta}$ and $\boldsymbol{z}$, respectively, in the same ordering of columns of $\tilde{G}$. Let $\tilde{C}$ be the code generated by $\tilde{G}$ which is equivalent to C. Define $\boldsymbol{u} = (u_1, u_2, \ldots, u_k) \in \{0,1\}^k$ as the leftmost k symbols of $\tilde{\boldsymbol{z}}$, i.e., $u_j = \tilde{z}_j, \forall j \in [1, k]$. The decoder first encodes $\boldsymbol{u}$ by $\tilde{G}$ to obtain the initial codeword $\tilde{\boldsymbol{c}}_{\emptyset} (= \boldsymbol{u}\tilde{G})$. Afterwards, k-dimensional vectors, called *test error patterns* $\boldsymbol{t} \in \{0,1\}^k$, are iteratively generated and encoded by $\tilde{G}$. Then, $\tilde{\boldsymbol{c}} = \tilde{\boldsymbol{c}}_{\emptyset} \oplus \boldsymbol{t}\tilde{G}$ is a candidate codeword[††]. This procedure is repeated until a sufficient condition for the optimality is satisfied.

Definition 1: For a position set $J \subseteq [1, k]$, the test error pattern (TEP) $\boldsymbol{t}(J) = \left(t_1, t_2, \ldots, t_k\right) \in \{0,1\}^k$ has element one in J and element zero in the complement of J. Such J is called the *support* of $\boldsymbol{t}(J)$. Define that $\mu(J)$ be the rightmost position in J, i.e., $\mu(J) = \max J$. For $j > \mu(J)$, the TEP $\boldsymbol{t}(J \cup \{j\})$ (or simply $\boldsymbol{t}(J \cup j)$) is called an *extended pattern* of $\boldsymbol{t}(J)$. For any J, define $J^{\mathrm{a}} = J \setminus \mu(J)$. For J and $\mu(J^{\mathrm{a}}) < j < \mu(J)$, the TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup j)$ is called an *adjacent pattern* of $\boldsymbol{t}(J)$ in j. □

Example 1: Assuming $k = 7$ and $J = \{2, 5\}$, then the TEP $\boldsymbol{t}(J)$ with the support J is $\boldsymbol{t}(J) = (0,1,0,0,1,0,0)$. Since $\mu(J) = 5$, there exist two extended patterns of $\boldsymbol{t}(J)$: $\boldsymbol{t}(J \cup 6)$ and $\boldsymbol{t}(J \cup 7)$. We find $J^{\mathrm{a}} = \{2\}$ and there also exist two adjacent patterns of $\boldsymbol{t}(J)$ in the position $j = 3, 4$: $\boldsymbol{t}(J^{\mathrm{a}} \cup 3) = (0,1,1,0,0,0,0)$ and $\boldsymbol{t}(J^{\mathrm{a}} \cup 4) = (0,1,0,1,0,0,0)$. □

For a binary vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in \{0,1\}^n$, we define the *correlation discrepancy* [8], [9] of $\boldsymbol{v}$ as

$$L(\boldsymbol{v}) = \sum_{j|\tilde{z}_j \neq v_j} |\tilde{\theta}_j|. \tag{1}$$

It is well-known that $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ is the most likely codeword if and only if $L(\tilde{\boldsymbol{c}}_{\mathrm{best}}) = \min_{\tilde{\boldsymbol{c}} \in \tilde{C}} L(\tilde{\boldsymbol{c}})$ [8], [10].

3. The Generalized BF Decoding Algorithm

3.1 Heuristic Functions of the Search

The methods considered in this paper generate TEPs according to their *heuristic values* (or *heuristics*). Here, we review heuristic functions which are used for searching the most likely codeword in [1]–[4], [8], [10].

Definition 2: For a TEP $\boldsymbol{t}(J)$, any function $F\left(\boldsymbol{t}(J)\right)$ satisfying

$$0 \leq F\left(\boldsymbol{t}(J)\right) \leq L(\tilde{\boldsymbol{c}}_J), \tag{2}$$

where $\tilde{\boldsymbol{c}}_J = (\tilde{c}_{J,1}, \tilde{c}_{J,2}, \ldots, \tilde{c}_{J,n}) = \tilde{\boldsymbol{c}}_{\emptyset} \oplus \boldsymbol{t}(J)\tilde{G}$, is called the *heuristic function* of the TEP. i.e., the heuristic value of $\boldsymbol{t}(J)$ is a lower bound of the discrepancy of $\tilde{\boldsymbol{c}}_J$. □

For a TEP $\boldsymbol{t}(J)$, the most simple heuristic function may be the correlation discrepancy over the k MRI positions defined as

$$\Delta\left(\boldsymbol{t}(J)\right) = \sum_{j \in J} |\tilde{\theta}_j|. \tag{3}$$

The function $\Delta(\cdot)$ actually satisfies Eq. (2), since $L(\tilde{\boldsymbol{c}}_J) = \Delta\left(\boldsymbol{t}(J)\right) + \sum_{j=k+1}^{n} (\tilde{z}_j \oplus \tilde{c}_{J,j}) |\tilde{\theta}_j|$. This heuristic function is used in [1], [4], [8], [10].

The heuristic function in [2], [3] utilizes the fact that any codeword in $\tilde{C}$ is at a distance $i \in W(\tilde{C})$ from a given codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$. For $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$ and $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$, we define[†††]

$$T\left(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right) = \Big\{ \boldsymbol{v} = (\boldsymbol{u} \oplus \boldsymbol{t}) \| (v_{k+1}, v_{k+2}, \ldots, v_n) \Big| \; d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) \in W(\tilde{C}) \Big\}, \tag{4}$$

where $d_H(\cdot, \cdot)$ represents the Hamming distance. If we do not know the exact weight profile $W(\tilde{C})$, which case is often occurred for long block codes, then we can substitute it by its superset. Then the heuristic function in [2], [3] is defined

[†]Since the probability of decision error of z_j becomes smaller as the value of $|\theta_j|$ is larger, $|\theta_j|$ is called reliability.

[††]The symbol $\oplus$ represents Exclusive OR operation.

[†††]The symbol $\|$ represents concatenation of vectors.

as

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = \sum_{j|\, t_j=1} |\tilde{\theta}_j| + \min_{\boldsymbol{v}\in T(\boldsymbol{t},\tilde{\boldsymbol{c}}_{\mathrm{ref}})} \Big\{ \sum_{j|\tilde{z}_j\neq v_j} |\tilde{\theta}_j| \Big\}. \tag{5}$$

Such $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ is called the *referenced codeword* [4], [5], [8] or the *seed* [2], [3], [6], [7].

We describe another heuristic function proposed by Fossorier and Lin [5]. For $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{C}$ and $\boldsymbol{t}$, we define

$$T_F\Big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\Big) = \Big\{ \boldsymbol{v} = (\boldsymbol{u} \oplus \boldsymbol{t}) \| (v_{k+1}, v_{k+2}, \ldots, v_n) \Big| \\ d_H(\boldsymbol{v}, \tilde{\boldsymbol{z}}) + d_H(\tilde{\boldsymbol{c}}_{\mathrm{ref}}, \tilde{\boldsymbol{z}}) \in W'(\tilde{C}) \Big\}, \tag{6}$$

where $W'(\tilde{C}) = \{0, d, d+1, \ldots, n\}$ is the superset of the weight profile $W(\tilde{C})$. Then the heuristic function in [5] is expressed as

$$g\Big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\Big) = \sum_{j|\, t_j=1} |\tilde{\theta}_j| + \min_{\boldsymbol{v}\in T_F(\boldsymbol{t},\tilde{\boldsymbol{c}}_{\mathrm{ref}})} \Big\{ \sum_{j|\tilde{z}_j\neq v_j} |\tilde{\theta}_j| \Big\}. \tag{7}$$

Note that since

$$d_H(\boldsymbol{v}, \tilde{\boldsymbol{z}}) = w_H(\boldsymbol{t}) + \#\{j|\tilde{z}_j \neq v_j\}, \tag{8}$$

the sequence $\boldsymbol{v} \in T_F\Big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\Big)$ which minimizes the second term of r.h.s. of Eq. (7) is determined only by the Hamming weight $w_H(\boldsymbol{t})$ [5], [9] where w_H and $\#\{\cdot\}$ represent the Hamming weight and the cardinality, respectively. In [5], Fossorier et al. have devised a method for making the function $g(\cdot)$ more effective according to updating the referenced codeword. For details, see [5].

The heuristic function is also used for reducing the time complexity of decoding procedure. Denote a currently best candidate codeword by $\tilde{\boldsymbol{c}}^*$. We note that if $F\Big(\boldsymbol{t}(J)\Big) \geq L(\tilde{\boldsymbol{c}}^*)$ for a TEP $\boldsymbol{t}(J)$, the candidate codeword $\tilde{\boldsymbol{c}}_J$ cannot be the most likely codeword because of Eq. (2). Hence, if all TEPs not encoded so far satisfy $F\Big(\boldsymbol{t}(J)\Big) \geq L(\tilde{\boldsymbol{c}}^*)$, then the sufficient condition for the optimality holds and we can terminate the decoding procedure. The tighter the lower bound of $L(\tilde{\boldsymbol{c}}_J)$ is, the more effective a sufficient condition for the optimality is. Since $f\Big(\boldsymbol{t}(J), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\Big) \geq \Delta\Big(\boldsymbol{t}(J)\Big)$ for any $\boldsymbol{t}(J)$, $f(\cdot)$ can give a tighter sufficient condition for the optimality than $\Delta(\cdot)$.

3.2 Generation Method of TEPs

We state how to dynamically generate TEPs according to their heuristic values. We will call the search strategies which process TEPs in the increasing order of their heuristic values the *priority-first search* [2], [3], [7].

The well-known MLD algorithm via the priority-first search is the A* decoding algorithm [2] in which the search is conducted by the A* algorithm through trellis or binary tree of the code. Although the A* decoding algorithm employs the function $f(\cdot)$, it performs the priority-first search with any heuristic functions $F(\cdot)$ satisfying the following condition:

(C1) $\quad F\Big(\boldsymbol{t}(J)\Big) \leq F\Big(\boldsymbol{t}(J \cup j)\Big) \quad \text{for } j \notin J.$

The heuristic functions $\Delta(\cdot)$, $f(\cdot)$ and $g(\cdot)$ actually satisfy the condition (C1) [9].

Other well-known MLD algorithm via the priority-first search is the BF decoding algorithm [1] which requires heuristic functions to satisfy not only the condition (C1) but also the condition:

$$F\Big(\boldsymbol{t}(J)\Big) \leq F\Big(\boldsymbol{t}(J')\Big) \\ \Rightarrow \quad F\Big(\boldsymbol{t}(J \cup j)\Big) \leq F\Big(\boldsymbol{t}(J' \cup j)\Big) \\ \text{for } j \notin J \cup J'. \tag{9}$$

It is readily shown that the function $\Delta(\cdot)$ satisfies Eq. (9) while the functions $f(\cdot)$ and $g(\cdot)$ do not necessarily satisfy it [8].

In [8], [9], Valembois et al. have shown that we can easily generalize the BF decoding algorithm to perform the priority-first search when the heuristic function satisfies only the condition (C1).

Hereafter, we assume heuristic functions satisfy (C1) and we will describe the GBF decoding algorithm. Let $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ represent k lists of TEPs. The TEP $\boldsymbol{t}(J)$ is supposed to be in $M^{(\mu(J))}$ where $\mu(J) = \max J$. Then the list for storing any TEP is uniquely determined. In a list $M^{(j)}, \forall j \in [1, k]$, TEPs are ordered in increasing order of their heuristic values. We call the TEP with the minimum heuristic value among all TEPs in lists the *best pattern*. The algorithm iteratively selects the best pattern, encodes it by $\tilde{G}$ and deletes it from lists. If there needs to generate new TEPs which have been not processed yet, the algorithm generates them. The basic strategy of generating TEPs is such that any TEP $\boldsymbol{t}(J)$ is not generated while we know that better patterns than $\boldsymbol{t}(J)$ are stored in lists or not generated so far.

In the initial stage of the algorithm, we construct the initial list of TEPs as follows: By the condition (C1), the TEPs with the minimum heuristic value in $M^{(j)}$, $j \in [1, k]$, is $\boldsymbol{t}(j)$ whose Hamming weight is one. i.e.,

$$\boldsymbol{t}(j) = \arg\min_{j=\mu(J)} \Big\{ F\Big(\boldsymbol{t}(J)\Big) \Big\} \tag{10}$$

for all $j \in [1, k]$. Therefore, we just need to set the initial lists as $M^{(j)} = \Big\{\boldsymbol{t}(j)\Big\}$ for $j \in [1, k]$. Thereafter, the algorithm selects the best patterns among TEPs that have not been processed[†].

We here describe the GBF decoding algorithm.

[The generalized BF decoding algorithm]

S1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs.

S2) Select the best pattern $\boldsymbol{t}(J) \in M^{(\mu(J))}$ among the topmost TEPs in non-empty lists $M^{(j)}$. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm.

[†]In [8], Valembois et al. have devised the technique for selecting the best pattern by $O(\lceil \log k \rceil)$ comparisons.

S3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_J$.
S4) For all lists $M^{(j)}$ such that $j > \mu(J)$, insert the extended patterns $\boldsymbol{t}(J \cup j)$ at the position such that the list remains increasing order of heuristic values. Delete $\boldsymbol{t}(J)$ from $M^{(\mu(J))}$.
S5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm. Otherwise, go to S2). □

In the above algorithm, S4) is the step of generating new TEPs which are extended patterns of $\boldsymbol{t}(J)$. We need to sort the generated TEP $\boldsymbol{t}(J \cup j)$ so that the list $M^{(j)}$ remains increasing order of the heuristic values. By sorting, the priority-first search is maintained.

The original BF decoding algorithm requires the heuristic functions to satisfy Eq. (9) as well as (C1). There we need not sort the new generated TEPs since Eq. (9) guarantees it is not better than any TEPs already stored in lists.

In the A* decoding algorithm, the only one list of TEPs is used. If we combine the k lists into the united list and order TEPs increasing order of heuristic values in it, then the above algorithm becomes identical to the A* decoding algorithm although the behaviors of the two algorithms seem different [8], [9]. Note that the essential properties are independent of the number of lists.

We here state the complexity of the GBF decoding algorithm. As for the space complexity, storing $\tilde{G}$ requires $O(kn)$ binary arrays. Denoting the maximum list size for decoding $\boldsymbol{r}$ by $M(\boldsymbol{r})$, the space complexity for lists is $O(k \times M(\boldsymbol{r}))$ binary arrays and $O(M(\boldsymbol{r}))$ arrays of real numbers. Therefore the overall space complexity is $O(\gamma)$ where $\gamma = \max\{kn, k \times M(\boldsymbol{r})\}$. If the maximum list size $M(\boldsymbol{r})$ is larger than n (which situations are usual from low to medium SNRs), the value $M(\boldsymbol{r})$ is dominant in the space complexity. It has been shown that the value $M(\boldsymbol{r})$ drastically increases as the SNR decreases.

As for the time complexity, permuting $\boldsymbol{\theta}$ in the non-increasing order of reliability costs $O(n \log n)$ comparisons and constructing $\tilde{G}$ costs $O(n \times \kappa^2)$ binary operations where $\kappa = \min\{k, n-k\}$ [2], [4], [5]. These steps are carried out only once in decoding of $\boldsymbol{r}$. Contrary to the above steps, generating $\boldsymbol{t}(J)$ and encoding them by $\tilde{G}$ are carried out iteratively, where each encoding requires $O(kn)$ binary operations by conventional encoding method [5], [6], [10]. For each TEP, computing its heuristic value costs real number operations of $O(n)$. Since a large number of TEPs are generated, both generating TEPs and the real number operations of heuristic values dominate mainly the whole decoding complexity [2], [6], [8] as well as encoding TEPs.

4. Proposed Decoding Algorithm

In this section, we propose a method for reducing the list size of TEPs in the GBF decoding algorithm. Before deriving the proposed method, we show some properties of conventional heuristic functions. These properties will be exploited by the proposed method.

4.1 Some Properties of Heuristic Functions

We here define the following condition for a heuristic function $F(\cdot)$.

Definition 3: Let $S^{(0)}$ be a certain subset of $[1, k]$ and $S^{(1)}$ be the complement of $S^{(0)}$. For $J \subseteq [1, k]$, assume $j_1, j_2 \notin J$ and $j_1 < j_2$. If $j_1, j_2 \in S^{(\alpha)}$ with $\alpha \in \{0, 1\}$, then a function $F(\cdot)$ satisfies

$$\textbf{(C2)} \qquad F\Big(\boldsymbol{t}(J \cup j_1)\Big) \geq F\Big(\boldsymbol{t}(J \cup j_2)\Big). \tag{11}$$

We will call this condition the condition (C2). □

The following propositions play important roles in deriving the improved method.

Proposition 1: Assume that $S^{(0)} = [1, k]$. Then the function $\Delta(\cdot)$ satisfies the condition (C2).

(**Proof**) Note that by Eq. (3), an extended pattern $\boldsymbol{t}(J \cup j)$ of $\boldsymbol{t}(J)$ such that $j \notin J$ satisfies

$$\Delta\Big(\boldsymbol{t}(J \cup j)\Big) = \Delta\Big(\boldsymbol{t}(J)\Big) + |\tilde{\theta}_j|. \tag{12}$$

If $1 \leq j_1 < j_2 \leq k$ and $j_2 \notin J$, then we have

$$\begin{aligned}&\Delta\Big(\boldsymbol{t}(J \cup j_1)\Big) - \Delta\Big(\boldsymbol{t}(J \cup j_2)\Big)\\ &\quad = \Delta\Big(\boldsymbol{t}(J)\Big) + |\tilde{\theta}_{j_1}| - \Delta\Big(\boldsymbol{t}(J)\Big) - |\tilde{\theta}_{j_2}| \geq 0\end{aligned} \tag{13}$$

since $|\tilde{\theta}_{j_1}| \geq |\tilde{\theta}_{j_2}|$. By the assumption, both j_1 and j_2 are in $S^{(0)} (= [1, k])$ and thus the function $\Delta(\cdot)$ satisfies (C2). □

Proposition 2: For a given referenced codeword $\tilde{\boldsymbol{c}}_{\rm ref} \in \tilde{C}$, let $\boldsymbol{t}_{\rm ref}$ be the TEP of $\tilde{\boldsymbol{c}}_{\rm ref}$ (i.e., $\tilde{\boldsymbol{c}}_{\rm ref} = \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}_{\rm ref}\tilde{G}$). Assuming that $S^{(1)}$ and $S^{(0)}$ be the support of $\boldsymbol{t}_{\rm ref}$ and its complement, respectively. Then the heuristic function $f(\cdot)$ satisfies the condition (C2).

In order to prove Proposition 2, we first show the following lemma.

Lemma 1: Denote the second term of the r.h.s. of Eq. (5) by $A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$, i.e.,

$$A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})} \Big\{ \sum_{j | \tilde{z}_j \neq v_j} |\tilde{\theta}_j| \Big\}. \tag{14}$$

Then for a given $\tilde{\boldsymbol{c}}_{\rm ref}$, any pairs $(\boldsymbol{t}, \boldsymbol{t}')$ such that $d_H(\boldsymbol{t}, \boldsymbol{t}_{\rm ref}) = d_H(\boldsymbol{t}', \boldsymbol{t}_{\rm ref})$ satisfy

$$A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = A(\boldsymbol{t}', \tilde{\boldsymbol{c}}_{\rm ref}) \tag{15}$$

i.e., the vector $\boldsymbol{v}$ which gives the minimum value of Eq. (14) is determined only by the Hamming distance $d_H(\boldsymbol{t}, \boldsymbol{t}_{\rm ref})$.

(**Proof**) By the definition, $\boldsymbol{v} \in T\Big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}\Big)$ satisfies

$$d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\rm ref}) = d_H(\boldsymbol{t}, \boldsymbol{t}_{\rm ref}) + \#\{j |\, v_j \neq \tilde{c}_{{\rm r},j}\} \tag{16}$$

where $\tilde{\boldsymbol{c}}_{\rm ref} = (\tilde{c}_{{\rm r},1}, \tilde{c}_{{\rm r},2}, \ldots, \tilde{c}_{{\rm r},n})$. In r.h.s., the first term expresses the distance over the left k positions and the second

term does the distance over the rest of $n-k$ positions.

For $\boldsymbol{t}$, if we assume that $\boldsymbol{v} = (\boldsymbol{u} \oplus \boldsymbol{t}) \| (v^*_{k+1}, v^*_{k+2}, \ldots, v^*_n)$ minimizes the r.h.s. of Eq. (14), then $\boldsymbol{v}$ must satisfy $d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\rm ref}) \in W(\tilde{C})$ from the condition in Eq. (4). Note that we have

$$A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = \sum_{j | v^*_j \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{17}$$

For another TEP $\boldsymbol{t}'$ such that $d_H(\boldsymbol{t}', \boldsymbol{t}_{\rm ref}) = d_H(\boldsymbol{t}, \boldsymbol{t}_{\rm ref})$, the vector $\boldsymbol{v}' = (\boldsymbol{u} \oplus \boldsymbol{t}') \| (v^*_{k+1}, v^*_{k+2}, \ldots, v^*_n)$ satisfies

$$d_H(\boldsymbol{v}', \tilde{\boldsymbol{c}}_{\rm ref}) = d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\rm ref}) \in W(\tilde{C}) \tag{18}$$

by Eq. (16). This equation implies $\boldsymbol{v}' \in T(\boldsymbol{t}', \tilde{\boldsymbol{c}}_{\rm ref})$ and

$$A(\boldsymbol{t}', \tilde{\boldsymbol{c}}_{\rm ref}) = \sum_{j | v'_j \neq \tilde{z}_j} |\tilde{\theta}_j|. \tag{19}$$

in which $\boldsymbol{v}'$ minimizes the r.h.s. of Eq. (14). Equations (17) and (19) complete the proof. □

(**Proof of Proposition 2**) Assuming that $j_1, j_2 \in S^{(0)}$ and $j_1, j_2 \notin J$, then the Hamming distance between TEPs $\boldsymbol{t}(J \cup j_1)$ and $\boldsymbol{t}_{\rm ref}$ and that between $\boldsymbol{t}(J \cup j_2)$ and $\boldsymbol{t}_{\rm ref}$ are the same since $S^{(0)}$ is the complement of the support of $\boldsymbol{t}_{\rm ref}$. i.e.,

$$d_H\Big(\boldsymbol{t}(J \cup j_1), \boldsymbol{t}_{\rm ref}\Big) = d_H\Big(\boldsymbol{t}(J \cup j_2), \boldsymbol{t}_{\rm ref}\Big). \tag{20}$$

Therefore, we have

$$A\Big(\boldsymbol{t}(J \cup j_1), \tilde{\boldsymbol{c}}_{\rm ref}\Big) = A\Big(\boldsymbol{t}(J \cup j_2), \tilde{\boldsymbol{c}}_{\rm ref}\Big) \tag{21}$$

from Lemma 1. Furthermore if $j_1 < j_2$, we have

$$\begin{aligned} &f\Big(\boldsymbol{t}(J \cup j_1), \tilde{\boldsymbol{c}}_{\rm ref}\Big) - f\Big(\boldsymbol{t}(J \cup j_2), \tilde{\boldsymbol{c}}_{\rm ref}\Big) \\ &\quad = \sum_{j \in J \cup j_1} |\tilde{\theta}_j| - \sum_{j \in J \cup j_2} |\tilde{\theta}_j| = |\tilde{\theta}_{j_1}| - |\tilde{\theta}_{j_2}| \geq 0. \end{aligned}$$

This inequality implies that the function $f(\cdot)$ satisfies (C2) if $j_1, j_2 \in S^{(0)}$. In the case of $j_1, j_2 \in S^{(1)}$, Eq. (20) also holds and we can prove the proposition similarly. □

Proposition 3: Assume that $S^{(0)} = [1, k]$. Then for a given $\tilde{\boldsymbol{c}}_{\rm ref}$, the function $g(\cdot)$ satisfies the condition (C2).

(**Proof**) Denote the second terms of r.h.s. of Eq. (7) by $B(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$ for a given $\boldsymbol{t}$. Recall that the vector $\boldsymbol{v} \in T_F(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$ which takes the value $B(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref})$ is determined only by $w_H(\boldsymbol{t})$ (see Sect. 3.1). i.e., arbitrary pairs $(\boldsymbol{t}, \boldsymbol{t}')$ with $w_H(\boldsymbol{t}) = w_H(\boldsymbol{t}')$ satisfy

$$B(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\rm ref}) = B(\boldsymbol{t}', \tilde{\boldsymbol{c}}_{\rm ref}). \tag{22}$$

Since $\boldsymbol{t}(J \cup j_1)$ and $\boldsymbol{t}(J \cup j_2)$ with $j_1, j_2 \notin J$ have the same Hamming weight, if $1 \leq j_1 < j_2 \leq k$, then

$$\begin{aligned} &g\Big(\boldsymbol{t}(J \cup j_1)\Big) - g\Big(\boldsymbol{t}(J \cup j_2)\Big) \\ &\quad = \Delta\Big(\boldsymbol{t}(J \cup j_1)\Big) - \Delta\Big(\boldsymbol{t}(J \cup j_2)\Big) \geq 0 \end{aligned} \tag{23}$$

where the last inequality is obtained from Eq. (13). By the assumption, both j_1 and j_2 must be in $S^{(0)} (= [1, k])$ and thus the function $g(\cdot)$ satisfies (C2). □

Propositions 1, 2 and 3 show that the heuristic functions $\Delta(\cdot)$, $f(\cdot)$ and $g(\cdot)$ satisfy the condition (C2) as well as (C1). In the following, we consider heuristic functions satisfying both (C1) and (C2).

4.2 Improved Generation Method of TEPs

In this section, we propose an improved method for reducing the list size of TEPs in the GBF decoding algorithm. For our purpose, we utilize the condition (C2) as well as (C1) to judge unnecessary TEPs and such unnecessary TEPs will not be generated as long as possible. More precisely, we regard a TEP $\boldsymbol{t}$ as *unnecessary* if it is clear that there is a TEP $\boldsymbol{t}'$ whose heuristic value is smaller than that of $\boldsymbol{t}$ in the lists. In the improved method, such an unnecessary TEP $\boldsymbol{t}$ is generated after the TEP $\boldsymbol{t}'$ is chosen as the best pattern at S2). This approach is similar to the improved technique for the original BF decoding algorithm† [8].

We also arrange k lists $M^{(j)}$ as in the GBF decoding algorithm. Hereafter, we denote $S^{(0)} = \{i_1, i_2, \ldots, i_s\}$ with $s \geq 1$ and $S^{(1)} = \{i'_1, i'_2, \ldots, i'_p\}$ with $p \geq 0$.

By the condition (C1), the TEP with the minimum heuristic value in a list $M^{(j)}$, $j \in [1, k]$, is $\boldsymbol{t}(j)$ whose Hamming weight is one. Furthermore, we can see that the best pattern among s TEPs $\boldsymbol{t}(j)$, $j \in S^{(0)}$, is $\boldsymbol{t}(i_s)$ by the condition (C2). Similarly, the best pattern among p TEPs $\boldsymbol{t}(j)$, $j \in S^{(1)}$, is $\boldsymbol{t}(i'_p)$. Therefore, we can construct the initial lists as

$$M^{(j)} = \begin{cases} \Big\{\boldsymbol{t}(j)\Big\}, & \text{if } j \in \{i_s, i'_p\}; \\ \emptyset, & \text{otherwise.} \end{cases} \tag{24}$$

Note that we generate at most two TEPs at this stage.

At S2) of the GBF decoding algorithm, if $\boldsymbol{t}(J) \in M^{(\mu(J))}$ is selected as the best pattern, $k - \mu(J)$ extended patterns of $\boldsymbol{t}(J)$ will be stored at S4). However, it is enough to store only its extended patterns $\boldsymbol{t}(J \cup i_s)$ and $\boldsymbol{t}(J \cup i'_p)$ in the list $M^{(i_s)}$ and $M^{(i'_p)}$, respectively. This is guaranteed by (C2), since

$$F\Big(\boldsymbol{t}(J \cup j)\Big) \geq F\Big(\boldsymbol{t}(J \cup i_s)\Big), \quad \text{for } \forall j \in S^{(0)} \tag{25}$$

$$F\Big(\boldsymbol{t}(J \cup j)\Big) \geq F\Big(\boldsymbol{t}(J \cup i'_p)\Big), \quad \text{for } \forall j \in S^{(1)} \tag{26}$$

where $\boldsymbol{t}(J \cup j)$ represents extended patterns of $\boldsymbol{t}(J)$.

Following this modification, we need to determine when to insert other extended patterns $\boldsymbol{t}(J \cup j)$, $j \notin \{i_s, i'_p\}$, into lists. Consider that a TEP $\boldsymbol{t}(J \cup i_q)$ such that $i_q \in S^{(0)}$ and $i_q > \mu(J)$ is stored in the list $M^{(i_q)}$. Since adjacent patterns $\boldsymbol{t}(J \cup j)$ such that $j \in S^{(0)}$ and $j < i_q$ cannot be the best pattern by (C2), we just need to store these adjacent patterns after $\boldsymbol{t}(J \cup i_q)$ is selected as the best pattern at S2). If $i_{q-1} > \mu(J)$, $\boldsymbol{t}(J \cup i_{q-1})$ has the smallest heuristic value

†Note again that the original BF decoding algorithm requires Eq. (9) for heuristic functions which is not satisfied by the function $f(\cdot)$ and $g(\cdot)$.

among all adjacent patterns of $\boldsymbol{t}(J \cup i_q)$ in $S^{(0)}$ from the condition (C2), i.e.,

$$\boldsymbol{t}(J \cup i_{q-1}) = \arg\min_{j \in S^{(0)}} \Big\{ F\big(\boldsymbol{t}(J \cup j)\big) \Big| \, j < i_q, j \notin J \Big\}. \quad (27)$$

Therefore, after $\boldsymbol{t}(J \cup i_q)$ is selected as the best pattern at S2), only $\boldsymbol{t}(J \cup i_{q-1})$ is inserted into the list $M^{(i_{q-1})}$. This modification significantly reduces the maximum list size. Similar arguments also hold when $\boldsymbol{t}(J \cup i'_q), i'_q \in S^{(1)}$, is selected as the best pattern at S2).

We describe a proposed decoding algorithm employing the above method.

[The proposed decoding algorithm]

P1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs by Eq. (24).

P2) Select the best pattern $\boldsymbol{t}(J) \in M^{(\mu(J))}$ among non-empty lists. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm.

P3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_J$.

P4) a) If $\mu(J) = i_q$ (i.e., $\mu(J) \in S^{(0)}$) and the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ exists where $J^{\mathrm{a}} = J \backslash \mu(J)$, then insert it into the list $M^{(i_{q-1})}$.

b) If $\mu(J) = i'_q$ (i.e., $\mu(J) \in S^{(1)}$) and the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ exists, then insert it into the list $M^{(i'_{q-1})}$.

c) If $\mu(J) < i_s$, then insert $\boldsymbol{t}(J \cup i_s)$ into $M^{(i_s)}$. If $\mu(J) < i'_p$, then insert $\boldsymbol{t}(J \cup i'_p)$ into $M^{(i'_p)}$. Delete $\boldsymbol{t}(J)$ from $M^{(\mu(J))}$.

P5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm. Otherwise, go to P2). □

The step P4) corresponds to the modification. Note that we need to store at most three TEPs at P4), while we need to store at most $k - \mu(J)$ TEPs at S4) of the GBF decoding algorithm.

Remark that we set $S^{(0)} = [1, k]$ by Propositions 1 and 3 if we employ either the function $\Delta(\cdot)$ or $g(\cdot)$. Since $S^{(1)} = \emptyset$, we can skip P4-b) and at most two TEPs (one is an adjacent pattern and the other is an extended pattern) are generated for each iteration (one iteration consists of selecting the best pattern, encoding it and generating new TEPs).

Example 2: Assuming $k = 7$ and $S^{(0)} = \{2, 4, 5\}$, let $\boldsymbol{t}(J) = (1, 0, 0, 1, 0, 0, 0)$ be the best pattern selected at P2). Since $\mu(J) = 4 \in S^{(0)}$, the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup 2) = (1, 1, 0, 0, 0, 0, 0)$ at the position $j = 2$ is inserted into the list $M^{(2)}$ at P4-a). Since $\mu(J) < i_s = 5$ and $\mu(J) < i'_p = 7$, extended patterns $\boldsymbol{t}(J \cup 5) = (1, 1, 0, 0, 1, 0, 0)$ and $\boldsymbol{t}(J \cup 7) = (1, 1, 0, 0, 0, 0, 1)$ of $\boldsymbol{t}(J)$ are inserted into the lists $M^{(5)}$ and $M^{(7)}$, respectively, at P4-c). □

We note that the next generated TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ at P4-a) can be easily computed from the selected best pattern $\boldsymbol{t}(J)(= \boldsymbol{t}(J^{\mathrm{a}} \cup i_q))$. Furthermore its heuristic value $F\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})\big)$ may be easily calculated from that of $\boldsymbol{t}(J)$. For example, if we adopt the function $\Delta(\cdot)$, the heuristic values of $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ is calculated as

$$\Delta\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})\big) = \Delta\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_q)\big) - |\tilde{\theta}_{i_q}| + |\tilde{\theta}_{i_{q-1}}|. \quad (28)$$

Similarly, if we adopt the function $f(\cdot)$, the heuristic values of $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), i_{q-1} \in S^{(0)}$, is calculated as

$$f\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) = f\big(\boldsymbol{t}(J^{\mathrm{a}} \cup i_q), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) - |\tilde{\theta}_{i_q}| + |\tilde{\theta}_{i_{q-1}}| \quad (29)$$

from Eq. (5) and Lemma 1. The heuristic value of $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1}), i'_{q-1} \in S^{(1)}$, can also be calculated by that of $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_q), i'_q \in S^{(1)}$, since the similar relationship as Eq. (29) holds between $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ and $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_q)$. As for the function $g(\cdot)$, if two TEPs have the same Hamming weight, the values $B(\cdot)$ of them take the same value. We can also calculate $f(\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1}), \tilde{\boldsymbol{c}}_{\mathrm{ref}})$ by the similar way of Eq. (29).

We show the validity of the proposed decoding algorithm.

Theorem 1: Assume that a heuristic function $F(\cdot)$ satisfies both (C1) and (C2). The ν-th iteration of the proposed decoding algorithm selects the TEP with the ν-th least heuristic value among all TEPs. i.e., it performs the priority-first search.

(**Proof**) See Appendix A. □

The foregoing theorem also implies that the proposed decoding algorithm performs MLD by Eq. (2).

Corollary 1: If we employ either $\Delta(\cdot)$, $f(\cdot)$ or $g(\cdot)$ as the heuristic function, the proposed decoding algorithm performs the priority-first search and achieves MLD. □

In terms of the space complexity of the proposed decoding algorithm, we show a lemma and a theorem.

Lemma 2: In each iteration of the proposed decoding algorithm, the list size of TEPs is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) From Theorem 1, both the GBF and the proposed decoding algorithms perform the priority-first search. Therefore, if we employ the same heuristic function, the numbers of TEPs selected as the best pattern at S2) and S4) (these numbers are equal to those of encoding TEPs) are identical.

Based on the above fact, we will prove the lemma by the mathematical induction. We denote the iteration number by ν.

(i) The case of $\nu = 1$:
The initial list constructed by Eq. (24) guarantees the list size of the proposed decoding algorithm is less than that in the GBF decoding algorithm.

(ii) The case of $\nu \geq 2$:
Assume that the list size in the $(\nu-1)$-th iteration of the proposed decoding algorithm is less than or equal to that

of the GBF decoding algorithm. When $\boldsymbol{t}(J)$ is selected as the best pattern in the ν-th iteration, all of its $k - \mu(J)$ extended patterns will be stored in lists at the step S4) in the GBF decoding algorithm, while at most two extended patterns of $\boldsymbol{t}(J)$ will be stored in lists at P4) of the same iteration in the proposed decoding algorithm. Furthermore, if proposed decoding algorithm needs to store the adjacent pattern of $\boldsymbol{t}(J)$, such adjacent pattern has been already stored in lists in the GBF decoding algorithm. Therefore, the list size in the ν-th iteration of the proposed decoding algorithm is less than or equal to that of the GBF decoding algorithm.

From the arguments (i) and (ii), we can prove the lemma. □

Theorem 2: The maximum list size of TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) We can readily prove the theorem by Lemma 2. □

We show the following theorem on the time complexity. Note that the number of TEPs generated in a decoding procedure is in general greater than the maximum list size and it is one of the indices to evaluate the time complexity of heuristic search MLD algorithms [2], [8].

Theorem 3: The number of generated TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2).

(**Proof**) We note again that if we employ the same heuristic function, the numbers of encoding TEPs for both decoding algorithms are identical. So both algorithms perform priority-first search and the number of iterations in which a sufficient condition for the optimality holds is the same. Furthermore, the TEPs generated in the ν-th iteration of the proposed decoding algorithm are obtained in the λ-th ($\lambda \leq \nu$) iteration of the GBF decoding algorithm. These facts guarantee that the number of generated TEPs in the proposed decoding algorithm is no more than that in the GBF decoding algorithm. □

5. Adaptive Procedures

5.1 Method for Updating Referenced Codeword

Some of the heuristic functions such as the functions $f(\cdot)$ and $g(\cdot)$ use a referenced codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$. In this case, we need not fix the referenced codeword throughout the decoding procedure. We call the decoding procedure in which the referenced codewords are updated the *adaptive procedure* [2], [6].

In [2], [5], [6], the currently best codeword $\tilde{\boldsymbol{c}}^*$ is set as referenced codeword. When a new (currently) best codeword $\tilde{\boldsymbol{c}}^*$ is obtained, the referenced codeword is updated by $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}^*$. Note that we initially set $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ in the first iteration since the first best codeword is necessarily $\tilde{\boldsymbol{c}}_{\emptyset}$. Since the number of TEPs stored in lists may be so large that we do not recalculate their heuristic values even when a referenced codeword is updated. If Eq. (2) holds for arbitrary pair of $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ and $\boldsymbol{t}(J)$, the GBF decoding algorithm still achieves MLD.

Let $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ denote the current referenced codeword in the ν-th iteration and $\tilde{\boldsymbol{c}}'_{\mathrm{ref}}$ be the referenced codeword for the selected best pattern $\boldsymbol{t}(J)$ at λ-th ($\lambda \leq \nu$) iteration. At S4) in the ν-th iteration of the GBF decoding algorithm, the heuristic value of $\boldsymbol{t}(J \cup j)$, an extended pattern of $\boldsymbol{t}(J)$, is calculated referring $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$. This $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ may differ from the referenced codeword $\tilde{\boldsymbol{c}}'_{\mathrm{ref}}$ which is referenced by $\boldsymbol{t}(J)$.

Meanwhile, in the ν-th iteration of the proposed decoding algorithm, we modify P4) as follows:

(a) When we obtain the adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ or $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ in P4-a) or P4-b) where $J^{\mathrm{a}} = J \setminus \mu(J)$, we calculate their heuristic values using $\tilde{\boldsymbol{c}}'_{\mathrm{ref}}$ which is the referenced codeword for the selected best pattern $\boldsymbol{t}(J)$.

(b) When we obtain the extended pattern $\boldsymbol{t}(J \cup i_s)$ or $\boldsymbol{t}(J \cup i'_p)$ in P4-c), we calculate their heuristic values using $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ which is the current referenced codeword.

Thus we need to store past referenced codewords $\tilde{\boldsymbol{c}}'_{\mathrm{ref}}$ in memory by above (a). However the increased space complexity may not be so large since the number of past referenced codewords is not so great compared to the list size of TEPs as we will see in the next section.

We have the following lemma on the proposed decoding algorithm with the adaptive procedure.

Lemma 3: For any TEP $\boldsymbol{t}(J)$, its heuristic value calculated in the proposed decoding algorithm is the same as that in the GBF decoding algorithm when we adopt the adaptive procedure.

(**Proof**) See Appendix B. □

Lemma 3 implies that the search order in the GBF and the proposed decoding algorithms with the adaptive procedures are strictly identical and they carry out MLD. Lemma 3 leads to the following theorems which are the counterparts of Theorems 2 and 3, respectively. The proofs are straightforward and hence we omit them.

Theorem 4: Assume that both the GBF and the proposed decoding algorithms employ the same heuristic function satisfying (C1) and (C2). Then the maximum list size of TEPs in the proposed decoding algorithm is less than that in the GBF decoding algorithm when they adopt the adaptive procedure.

Theorem 5: Assume that both the GBF and the proposed decoding algorithms employ the same heuristic function satisfying (C1) and (C2). Then the number of generated TEPs in the proposed decoding algorithm is no more than that in

the GBF decoding algorithm when they adopt the adaptive procedure.

5.2 The Case of Specific Heuristic Functions

If we use the heuristic function $f(\cdot)$, we can reduce the increased (time and space) complexity in the proposed decoding algorithm with the adaptive procedure (the increased space complexity is required to store past referenced codewords). For the TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup i_{q-1})$ generated at P4-a), since its referenced codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ is the same as that for the selected best pattern $\boldsymbol{t}(J)(= \boldsymbol{t}(J^{\mathrm{a}} \cup i_q))$, its heuristic value can be calculated by Eq. (29). To calculate r.h.s. of Eq. (29), we just need to know the value $f\left(\boldsymbol{t}(J), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\right)$ and the position $i_{q-1} \in S^{(0)}$ which is the adjacent to $i_q \in S^{(0)}$. For this reason, we need to store TEPs (not codewords themselves) corresponding to old referenced codewords in memory. We call these TEPs *referenced TEPs*. The similar argument holds for the TEP $\boldsymbol{t}(J^{\mathrm{a}} \cup i'_{q-1})$ generated at P4-b) where $i'_{q-1} \in S^{(1)}$. We remark that the space complexity for storing referenced TEPs are smaller than that for storing ordinary TEPs since we need not store heuristic values of referenced TEPs.

If we use the heuristic function $g(\cdot)$, we can further save the space complexity for storing the past referenced codeword. As we see in Sect. 3.1, the value $B(\cdot)$ of a TEP defined in Eq. (22) depends only on its Hamming weight. Therefore even if the referenced codeword is updated in the adaptive procedure, we need not hold the past referenced codewords since we can calculate the heuristic value of the generated adjacent pattern by its Hamming weight at P4-a). There is no increased space complexity compared to the GBF decoding algorithm.

6. Simulation Results

In this section, we evaluate the effectiveness of the proposed decoding algorithm by computer simulations.

6.1 Conditions of Simulations

For the binary (63, 30, 13) BCH code and the binary (104, 52, 20) quadratic residue (QR) code, we perform MLD by the GBF decoding algorithm (we denote it by "GBF" in tables) and the proposed decoding algorithm (we denote it by "Proposed" in tables). At each SNR E_{b}/S_0 [dB], both decoding algorithms are carried out 10,000 times.

We adopt the function $f(\cdot)$ as the heuristic function in both decoding algorithms. We remark again that this GBF decoding algorithm is identical to the well-known A* decoding algorithm [2]. We assume that the weight profiles $W(C)$ of these two codes are unknown and we use their supersets $W'(C) = \{0, d, d{+}1, \cdots, n\}$. We compare two versions according to the way of arranging the referenced codeword: (i) We fix the referenced codeword as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ and we set $S^{(0)} = [1, k]$ and $S^{(1)} = \emptyset$. So we can skip P4-b) of the proposed decoding algorithm and the new extended pattern which is generated at P4) is only one. (ii) We consider the adaptive procedure in which the referenced codeword is initially set as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ and then updated as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}^*$ each time a new (currently) best codeword is obtained.

In tables, we use the following notations:

- $N(\boldsymbol{r})$: the number of generated TEPs in decoding of $\boldsymbol{r}$
- $M(\boldsymbol{r})$: the maximum list size in decoding of $\boldsymbol{r}$
- $R(\boldsymbol{r})$: the number of updating referenced codeword in the proposed decoding algorithm
- Ave : the average value among 10,000 times of decoding
- Max : the maximum value among 10,000 times of decoding

6.2 Results and Discussion

(The results on the space complexity for algorithms fixing $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$)

We show the results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for the (63, 30, 13) BCH code and the (104, 52, 20) QR code in Tables 1 and 2, respectively.

By Table 1, the maximum list size Max $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/3 of that in the GBF decoding algorithm at each SNR. Furthermore, the average

Table 1 The results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for (63, 30, 13) BCH code.

E_{b}/N_0 [dB]			GBF	Proposed
5.0	Ave	$N(\boldsymbol{r})$	$3.54 \cdot 10^1$	2.10
		$M(\boldsymbol{r})$	1.46	$1.92 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$2.073 \cdot 10^3$	$4.110 \cdot 10^2$
4.0	Ave	$N(\boldsymbol{r})$	$1.09 \cdot 10^2$	$2.50 \cdot 10^1$
		$M(\boldsymbol{r})$	$1.40 \cdot 10^1$	2.28
	Max	$M(\boldsymbol{r})$	$9.586 \cdot 10^3$	$2.406 \cdot 10^3$
3.0	Ave	$N(\boldsymbol{r})$	$6.74 \cdot 10^2$	$2.33 \cdot 10^2$
		$M(\boldsymbol{r})$	$1.13 \cdot 10^2$	$2.31 \cdot 10^1$
	Max	$M(\boldsymbol{r})$	$1.582 \cdot 10^4$	$4.600 \cdot 10^3$
2.0	Ave	$N(\boldsymbol{r})$	$3.15 \cdot 10^3$	$1.26 \cdot 10^3$
		$M(\boldsymbol{r})$	$5.82 \cdot 10^2$	$1.36 \cdot 10^2$
	Max	$M(\boldsymbol{r})$	$7.052 \cdot 10^4$	$2.055 \cdot 10^4$

Table 2 The results of decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}_{\emptyset}$ for $(104, 52, 20)$ QR code.

E_{b}/N_0 [dB]			GBF	Proposed
6.0	Ave	$N(\boldsymbol{r})$	$4.94 \cdot 10^1$	$4.18 \cdot 10^{-1}$
		$M(\boldsymbol{r})$	$3.68 \cdot 10^{-1}$	$3.26 \cdot 10^{-2}$
	Max	$M(\boldsymbol{r})$	$5.160 \cdot 10^2$	$4.400 \cdot 10^1$
5.0	Ave	$N(\boldsymbol{r})$	$1.18 \cdot 10^2$	$1.27 \cdot 10^1$
		$M(\boldsymbol{r})$	7.46	$7.65 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$1.093 \cdot 10^4$	$1.264 \cdot 10^3$
4.0	Ave	$N(\boldsymbol{r})$	$2.24 \cdot 10^3$	$5.98 \cdot 10^2$
		$M(\boldsymbol{r})$	$2.89 \cdot 10^2$	$4.68 \cdot 10^1$
	Max	$M(\boldsymbol{r})$	$4.634 \cdot 10^5$	$9.875 \cdot 10^4$
3.0	Ave	$N(\boldsymbol{r})$	$4.70 \cdot 10^4$	$1.32 \cdot 10^4$
		$M(\boldsymbol{r})$	$7.38 \cdot 10^3$	$1.16 \cdot 10^3$
	Max	$M(\boldsymbol{r})$	$1.145 \cdot 10^7$	$2.674 \cdot 10^6$

Table 3 The results of decoding with adaptive procedure for (63, 30, 13) BCH code.

E_b/N_0 [dB]			GBF	Proposed
5.0	Ave	$N(\boldsymbol{r})$	$3.37 \cdot 10^1$	1.78
		$M(\boldsymbol{r})$	1.19	$1.76 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$2.064 \cdot 10^3$	$4.370 \cdot 10^2$
4.0	Ave	$N(\boldsymbol{r})$	$9.36 \cdot 10^1$	$2.19 \cdot 10^1$
		$M(\boldsymbol{r})$	$1.14 \cdot 10^1$	2.19
	Max	$M(\boldsymbol{r})$	$9.579 \cdot 10^3$	$3.272 \cdot 10^3$
3.0	Ave	$N(\boldsymbol{r})$	$5.83 \cdot 10^2$	$2.25 \cdot 10^2$
		$M(\boldsymbol{r})$	$9.66 \cdot 10^1$	$2.44 \cdot 10^1$
	Max	$M(\boldsymbol{r})$	$1.432 \cdot 10^4$	$5.103 \cdot 10^3$
2.0	Ave	$N(\boldsymbol{r})$	$2.91 \cdot 10^3$	$1.31 \cdot 10^3$
		$M(\boldsymbol{r})$	$5.53 \cdot 10^2$	$1.56 \cdot 10^2$
	Max	$M(\boldsymbol{r})$	$7.052 \cdot 10^4$	$2.801 \cdot 10^4$

Table 4 The results of decoding with adaptive procedure for (104, 52, 20) QR code.

E_b/N_0 [dB]			GBF	Proposed
6.0	Ave	$N(\boldsymbol{r})$	$4.85 \cdot 10^1$	$2.39 \cdot 10^{-1}$
		$M(\boldsymbol{r})$	$2.48 \cdot 10^{-1}$	$3.42 \cdot 10^{-2}$
	Max	$M(\boldsymbol{r})$	$1.87 \cdot 10^2$	$3.20 \cdot 10^1$
5.0	Ave	$N(\boldsymbol{r})$	$8.24 \cdot 10^1$	5.79
		$M(\boldsymbol{r})$	3.60	$4.71 \cdot 10^{-1}$
	Max	$M(\boldsymbol{r})$	$4.932 \cdot 10^3$	$7.930 \cdot 10^2$
4.0	Ave	$N(\boldsymbol{r})$	$1.23 \cdot 10^3$	$3.57 \cdot 10^2$
		$M(\boldsymbol{r})$	$1.66 \cdot 10^2$	$3.01 \cdot 10^1$
	Max	$M(\boldsymbol{r})$	$4.630 \cdot 10^5$	$9.862 \cdot 10^4$
3.0	Ave	$N(\boldsymbol{r})$	$3.38 \cdot 10^4$	$1.30 \cdot 10^4$
		$M(\boldsymbol{r})$	$5.46 \cdot 10^3$	$1.30 \cdot 10^3$
	Max	$M(\boldsymbol{r})$	$1.145 \cdot 10^7$	$2.681 \cdot 10^6$

value of the maximum list size Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/4 of that in the GBF decoding algorithm. These results show that the effectiveness of the proposed decoding algorithm. By Table 2, the values Max $M(\boldsymbol{r})$ and Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 and 1/6 of those in the GBF decoding algorithm, respectively. These results indicate that the proposed method also works well for the (104, 52, 20) QR code.

(The results on the space complexity for the adaptive procedure)

We show the results of decoding with the adaptive procedure (in which we update as $\tilde{\boldsymbol{c}}_{\mathrm{ref}} = \tilde{\boldsymbol{c}}^*$) for the (63, 30, 13) BCH code and the (104, 52, 20) QR code in Tables 3 and 4, respectively. We also show the number of updating referenced codeword (which is equal to the number of the past referenced TEPs stored in memory) in the proposed decoding algorithm with the adaptive procedure in Tables 5 and 6.

By Table 3 for the (63, 30, 13) BCH code, the maximum list size Max $M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 2/5 of that in the GBF decoding algorithm at each SNR. Furthermore, the average values of the maximum list size Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less

Table 5 The number of past referenced codewords in the proposed decoding algorithm for the (63, 30, 13) BCH code.

E_b/N_0	Ave $R(\boldsymbol{r})$	Max $R(\boldsymbol{r})$
5.5	1.025	10
5.0	1.056	12
4.5	1.117	12
4.0	1.215	14
3.5	1.394	16
3.0	1.638	18
2.5	1.984	16
2.0	2.425	24

Table 6 The number of past referenced codewords in the proposed decoding algorithm for the (104, 52, 20) QR code.

E_b/N_0	Ave $R(\boldsymbol{r})$	Max $R(\boldsymbol{r})$
6.5	1.004	8
6.0	1.013	8
5.5	1.039	10
5.0	1.101	12
4.5	1.209	14
4.0	1.393	16
3.5	1.714	20
3.0	2.189	22

than 1/3 of those in the GBF decoding algorithm. By Table 4 for the (104, 52, 20) QR code, the values Max $M(\boldsymbol{r})$ and Ave $M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 of those in the GBF decoding algorithm[†]. By Tables 5 and 6, the average values of $R(\boldsymbol{r})$ are fairly small and the maximum value of $R(\boldsymbol{r})$ is only 24 at 2.0 [dB] for the (63, 30, 13) BCH code. On the other hand, the average and the maximum values of $M(\boldsymbol{r})$ are 156 and 2,801 at 2.0 [dB], respectively, so the values $R(\boldsymbol{r})$ seem to be negligible. These values demonstrate that there are almost no increases of the space complexity for the proposed decoding algorithm even though we store the past referenced TEPs. Note that the average and maximum values $R(\boldsymbol{r})$ are hardly increased even for the long (104, 52, 20) QR code.

(The results on the number of generating TEPs)

The number of generating TEPs $N(\boldsymbol{r})$ is one of indices to evaluate time complexity in heuristic search MLD algorithms [2], [8] although the reduction of the time complexity led by reducing $N(\boldsymbol{r})$ may not be so large in the whole decoding complexity.

By Tables 1 and 2 for decoding with fixed $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$, $N(\boldsymbol{r})$ in the proposed decoding algorithm are less than 2/5 of $N(\boldsymbol{r})$ in the GBF decoding algorithm even at low SNRs. These results demonstrate the proposed decoding algorithm reduces the time complexity of the GBF decoding algorithm as well as the space complexity.

By Tables 3 and 4 for decoding algorithms with adap-

[†]The ratio of the value Ave $M(\boldsymbol{r})$ of the method in [6] to that of the GBF decoding algorithm is about 2/5 at 5.0 [dB] for the (104, 52, 20) QR code. By Table 4, the ratio of the value Ave $M(\boldsymbol{r})$ of the proposed decoding algorithm to that of the GBF decoding algorithm is about 1/8 at 5.0 [dB] for the (104, 52, 20) QR code .

tive procedure, $N(\boldsymbol{r})$ in the proposed decoding algorithm is less than 2/5 of $N(\boldsymbol{r})$ in the GBF decoding algorithm even at 3.0 [dB]. These results demonstrate the proposed method also reduces the time complexity of the GBF decoding algorithm even when we adopt the adaptive procedure.

7. Concluding Remarks

In this paper, we propose a new heuristic search method for reducing the space complexity of the GBF decoding algorithm. The GBF decoding algorithm is identical to the well-known A* decoding algorithm and includes the original BF decoding algorithm. As a result, the proposed method reduces the space complexity of the well-known A* and the original BF decoding algorithms. Though heuristic functions considered here are restricted by a condition, we show this class of heuristic functions includes some well-known functions. The proposed decoding algorithm guarantees to perform MLD since the set of generated candidate codewords is identical to that in the GBF decoding algorithm. Since the proposed decoding algorithm also reduces the number of generated TEPs which tends to vastly increase from low to medium SNRs, the proposed decoding algorithm reduces not only the space complexity but the time one in the GBF decoding algorithm.

As future works, we need to develop a method for heuristic search MLD algorithm with powerful heuristic functions such as in [6], [7]. More detailed comparisons between the proposed decoding algorithm and methods in [6], [7] are to be evaluated.

Acknowledgments

H. Yagi wishes to thank Dr. M. Kobayashi at Shonan Institute of Technology and Mr. T. Ishida and Mr. G. Hosoya at Waseda University for their valuable comments.

References

[1] G. Battail and J. Fang, "Décodage pondéré optimal descodes linéaires en blocs," Annales des télé-communications, vol.41, nos.11–12, pp.580–604, Nov.–Dec. 1986.
[2] Y.S. Han, C.R.P. Hartmann, and C.C. Chan, "Efficient priority-first search maximum likelihood soft decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.39, no.5, pp.1514–1523, Sept. 1993.
[3] Y.S. Han, C.R.P. Hartmann, and K.G. Mehrotra, "Decoding linear block codes using a priority-first search: Performance analysis and suboptimal version," IEEE Trans. Inf. Theory, vol.44, no.3, pp.1233–1246, May 1998.
[4] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," IEEE Trans. Inf. Theory, vol.43, no.1, pp.239–249, Jan. 1997.
[5] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," IEEE Trans. Inf. Theory, vol.41, no.5, pp.1379–1396, Sept. 1995.
[6] T. Okada, M. Kobayashi, and S. Hirasawa, "An efficient heuristic search method for maximum likelihood decoding of linear block codes using dual codes," IEICE Trans. Fundamentals, vol.E85-A, no.2, pp.485–489, Feb. 2002.
[7] C.C. Shih, C.R. Wulff, C.R.P. Hartmann, and C.K. Mohan, "Efficient heuristic search algorithms for soft-decision decoding of linear block codes," IEEE Trans. Inf. Theory, vol.44, no.7, pp.3023–3038, Nov. 1998.
[8] A. Valembois and M. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application to soft decision decoding," IEEE Commun. Lett., vol.5, no.3, pp.456–458, Nov. 2001.
[9] A. Valembois and M. Fossorier, "A comparison between "most-reliable-basis reprocessing" strategies," IEICE Trans. Fundamentals, vol.E85-A, no.7, pp.1727–1741, July 2002.
[10] H. Yagi, M. Kobayashi, T. Matsushima, and S. Hirasawa, "Complexity reduction of the Gazelle and Snyders decoding algorithm for maximum likelihood decoding," IEICE Trans. Fundamentals, vol.E86-A, no.10, pp.2461–2472, Oct. 2003.
[11] H. Yagi, T. Matsushima, and S. Hirasawa, "A method for reducing space complexity of reliability-based heuristic search maximum likelihood decoding algorithms," Proc. 26th Symposium on Information Theory and its Applications (SITA2003), pp.185–188, Hyogo, Japan, Dec. 2003.
[12] H. Yagi, T. Matsushima, and S. Hirasawa, "A heuristic search algorithm with the reduced list of test error patterns for maximum likelihood decoding algorithms," Proc. 27th Symposium on Information Theory and its Applications (SITA2004), pp.571–574, Gifu, Japan, Dec. 2004.

Appendix A: The Proof of Theorem 1

It is sufficient to show that the TEP with the ν-th smallest heuristic value among all TEPs has been already generated and stored in the list at the beginning of ν-th iteration of the decoding algorithm. We will prove it by the mathematical induction. Let $\boldsymbol{t}(J_q)$ with the support J_q be the TEP with the q-th smallest heuristic value among all possible TEPs.

(i) The case of $\nu = 1$:
By the conditions (C1) and (C2), the best pattern $\boldsymbol{t}(J_1)$ is either $\boldsymbol{t}(i_s)$ or $\boldsymbol{t}(i'_p)$. The initial list of TEPs is constructed by Eq. (24) so $\boldsymbol{t}(J_1)$ has been already stored in the list $M^{(\mu(J_1))}$ in the first iteration.

(ii) The case of $\nu > 1$:
Let $\mathcal{T}_\nu$ denote the set of all $\nu - 1$ best patterns before the ν-th iteration. i.e.,

$$\mathcal{T}_\nu = \left\{ \boldsymbol{t}(J_q) \,|\, q = 1, 2, \ldots, \nu - 1 \right\}. \tag{A·1}$$

Assume that we have exactly selected all $\nu - 1$ TEPs in $\mathcal{T}_\nu$ before the ν-th iteration.

(a) If $\mu(J_\nu) = i_q \in S^{(0)}$ such that $i_q < i_s$, there is $\boldsymbol{t}(J)$ such that both $\mu(J) = i_{q+1}$ and $\boldsymbol{t}(J_\nu)$ is the adjacent pattern of $\boldsymbol{t}(J)$. We have $F(\boldsymbol{t}(J)) \le F(\boldsymbol{t}(J_\nu))$ and $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ by the condition (C2). Therefore $\boldsymbol{t}(J)$ was selected as the best pattern at P2) in a previous iteration. When such $\boldsymbol{t}(J)$ was selected as the best pattern, $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i_q)}$ at P4-a) in the same iteration.

(b) Arguing similarly to the case of $\mu(J_\nu) \in S^{(0)}$, when $\mu(J_\nu) = i'_q \in S^{(1)}$ and $i'_q < i'_p$, $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i'_q)}$ at P4-b) in a previous iteration.

(c) If $\mu(J_\nu) = i_s \in S^{(0)}$, the TEP $\boldsymbol{t}(J)$ such that $J = J_\nu \backslash \{i_s\}$ satisfies $F(\boldsymbol{t}(J)) \le F(\boldsymbol{t}(J_\nu))$ and $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ by the condition (C1). Therefore such $\boldsymbol{t}(J) \in \mathcal{T}_\nu$ was selected

as the best pattern at P2) in a previous iteration and then $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i_s)}$ at P4-c) in the same iteration.

Similarly, when $\mu(J_\nu) = i'_q \in S^{(1)}$, we can show $\boldsymbol{t}(J_\nu)$ has inserted into the list $M^{(i'_p)}$ at P4-c) in the λ-th iteration such that $\lambda \leq \nu$.

As we mentioned in (i), since the first best pattern $\boldsymbol{t}(J_1)$ has been generated when initial lists has been constructed by Eq. (24), the assumptions of (ii) are satisfied and this completes the proof. □

Appendix B: The Proof of Lemma 3

We will prove the lemma by the mathematical induction. Let ν represent the iteration number.

(i) The case of $\nu = 1$:
The first referenced codeword is the same ($\tilde{\boldsymbol{c}}_{\text{ref}} = \tilde{\boldsymbol{c}}_\emptyset$) in both decoding algorithm. So the heuristic values of the first best pattern selected at S2) and P2) are identical.

(ii) The case of $\nu \geq 2$:
Assume that heuristic values of all TEPs generated before the ν-th iteration of the proposed decoding algorithm are the same as those of the GBF decoding algorithm. So the heuristic values of the best pattern $\boldsymbol{t}(J)$ selected at S2) and P4) in the ν-the iteration are identical.
We first consider the heuristic value of $\boldsymbol{t}(J^{\text{a}} \cup i_q), i_q \in S^{(0)}$, which is obtained at P4-a) after selecting $\boldsymbol{t}(J)$ as the best pattern. This TEP $\boldsymbol{t}(J^{\text{a}} \cup i_q)$ is generated at the same iteration of generating $\boldsymbol{t}(J)(= \boldsymbol{t}(J^{\text{a}} \cup i_{q+1})), i_{q+1} \in S^{(0)}$, in the GBF decoding algorithm since both of them are extended patterns of $\boldsymbol{t}(J^{\text{a}})$. Then their heuristic values are calculated by referring the same codeword, say $\tilde{\boldsymbol{c}}'_{\text{ref}}$. On the other hand, by the step (a) of the proposed decoding algorithm with the adaptive procedure, if we calculate the heuristic value of $\boldsymbol{t}(J^{\text{a}} \cup i_q)$ by referring $\tilde{\boldsymbol{c}}'_{\text{ref}}$, it is identical to that in the GBF decoding algorithm. Similarly, the heuristic value of $\boldsymbol{t}(J^{\text{a}} \cup i'_q), i'_q \in S^{(1)}$, which is obtained at P4-b) is calculated by the same referenced codeword and thus it has the same heuristic value in both decoding algorithm.
Next we consider the heuristic values of $\boldsymbol{t}(J \cup i_s)$ and $\boldsymbol{t}(J \cup i'_p)$ which are obtained at P4-c) after selecting $\boldsymbol{t}(J)$ as the best pattern. These TEPs $\boldsymbol{t}(J \cup i_s)$ and $\boldsymbol{t}(J \cup i'_p)$ are also generated in the same iteration of the GBF decoding algorithm and hence the heuristic values of them are identical. Therefore, heuristic values of TEPs generated in the ν-th iteration of the proposed decoding algorithm are the same as those of the GBF decoding algorithm.

The arguments (i) and (ii) complete the proof. □

Appendix C: Comparison with the Improved Method of [8]

Valembois et al. have proposed an improved method of the original BF decoding algorithm (we will call this method the improved BF (IBF) decoding algorithm) [8]. In this section, we compare the proposed decoding algorithm with the IBF decoding algorithm.

The IBF decoding algorithm exploits the property of the function $\Delta(\cdot)$ satisfying Eq. (9) as well as the condition (C1). We will consider any heuristic functions $F(\cdot)$ satisfying both the condition (C1) and Eq. (9).

In the IBF decoding algorithm, each list $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ contains at most one TEP while we arrange another list A which stores TEPs already selected as the best pattern at S2). After a TEP $\boldsymbol{t}(J)$ is selected as the best pattern, it is added to the end of the list A.

The initial lists of TEPs are constructed by

$$M^{(j)} = \{\boldsymbol{t}(j)\} \qquad \text{for } j \in [1, k], \tag{A·2}$$

as in the original BF decoding algorithm. In the initial step of the algorithm, the list A is set as $A = \emptyset$.

When a TEP $\boldsymbol{t}(J), \mu(J) \neq k$, is selected as the best pattern, we delete it from the list $M^{(\mu(J))}$ and added it to the end of the list A. It is readily shown by Eq. (9) that the next TEP to be stored in the list $M^{(\mu(J))}$ is $\boldsymbol{t}(J' \cup \mu(J))$ where $\boldsymbol{t}(J')$ is given by

$$\boldsymbol{t}(J') = \arg\min_{\boldsymbol{t}(I)} \left\{ F(\boldsymbol{t}(I)) \geq F(\boldsymbol{t}(J^{\text{a}})) \middle| \mu(I) < \mu(J) \right\}. \tag{A·3}$$

We can show that the $\boldsymbol{t}(J')$ is the first TEP with $\mu(J') < \mu(J)$ following $\boldsymbol{t}(J^{\text{a}})$ in the list A if it has been already stored in the list A. Note that if a selected best pattern has no extended patterns or if all its extended patterns have been already generated, we do not need to possess it in the list A.

[The improved BF decoding algorithm [8]**]**

S'1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs by Eq. (A·2).

S'2) Select the best pattern $\boldsymbol{t}(J) \in M^{(\mu(J))}$ among TEPs in non-empty lists $M^{(j)}$. If $F(\boldsymbol{t}(J)) \geq \underline{L}$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm.

S'3) Generate the next candidate codeword by $\tilde{\boldsymbol{c}}_J := \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$. If $L(\tilde{\boldsymbol{c}}_J) < \underline{L}$, then set $\underline{L} := L(\tilde{\boldsymbol{c}}_J)$ and $\tilde{\boldsymbol{c}}^* := \tilde{\boldsymbol{c}}_J$.

S'4) a) Delete $\boldsymbol{t}(J)$ from the list $M^{(\mu(J))}$. If $\mu(J) \neq k$, store $\boldsymbol{t}(J)$ into the list A.

b) Find a TEP $\boldsymbol{t}(J')$ which satisfies Eq. (A·3) in the list A. If such $\boldsymbol{t}(J')$ exists, then generate $\boldsymbol{t}(J' \cup \mu(J))$ and store it in the list $M^{(\mu(J))}$.

c) If there is an empty list $M^{(j)}$ with $\mu(J^{\text{a}}) < j$, $j \neq \mu(J)$, store $\boldsymbol{t}(J^{\text{a}} \cup j)$ in the list $M^{(j)}$.

d) If all extended patterns of the TEP $\boldsymbol{t}(J^{\text{a}})$ have been already generated, delete it from the list A.

S'5) If $M^{(j)} = \emptyset$ for all $j \in [1, k]$, then output $\tilde{\boldsymbol{c}}^*$ and halt the algorithm. Otherwise, go to S2). □

Note that for a selected best pattern $\boldsymbol{t}(J)$, at most one TEP is newly stored in a list (the selected best pattern is only moved from the list $M^{(\mu(J))}$ to the list A and the new generated TEP is stored in the list $M^{(\mu(J))}$). Therefore the

list size increases at most by one in each iteration.

We can easily show that the function $\Delta(\cdot)$ satisfies conditions both (C1) and Eq. (9). Since the function $\Delta(\cdot)$ also satisfies the condition (C2), the proposed decoding algorithm can also employ it. By Proposition 1, we set $S^{(0)} = [1, k]$ and the number of TEPs newly stored in lists in each iteration of the proposed decoding algorithm is at most two (one is an extended pattern and the other is an adjacent pattern). The list size of the proposed decoding algorithm also increases at most by one in each iteration since the selected best pattern is deleted from the list.

We have the following proposition on the relationship between the IBF decoding algorithm and the proposed decoding algorithm.

Proposition 4: Assume that both the IBF and the proposed decoding algorithms employ a heuristic function satisfying the condition (C1) and Eq. (9). We further assume that the heuristic function satisfies the condition (C2) with $S^{(0)} = [1, k]$ such as the function $\Delta(\cdot)$. Then the maximum list size of TEPs in the proposed decoding algorithm is no more than that in the IBF decoding algorithm.

(**Proof**) We will prove the proposition by mathematical induction. We here denote the iteration number by ν.

(i) The case of $\nu = 1$:
The initial list constructed by Eqs. (24) and (A· 2) indicate the list size of the proposed decoding algorithm is no more than that in the IBF decoding algorithm. Note that Eq. (9) cannot tell that we only need to store $\boldsymbol{t}(k)$ in lists so we need to store other TEPs with Hamming weight one in the IBF decoding algorithm.

(ii) The case of $\nu \geq 2$:
We first remark that the selected best pattern $\boldsymbol{t}(J)$ in the ν-th iteration of the IBF and the proposed decoding algorithms is identical since both algorithms perform the priority-first search.
Denote the set of TEPs stored in lists in the proposed and the IBF decoding algorithms by $\mathcal{T}_p$ and $\mathcal{T}_{IBF}$, respectively. Then there exists a one-to-one mapping ϕ from each element of $\mathcal{T}_p$ to that of $\mathcal{T}_{IBF}$ given by

$$\phi : \mathcal{T}_p \to \mathcal{T}_{IBF}, \tag{A· 4}$$

and

$$\phi(\boldsymbol{t}) \neq \phi(\boldsymbol{t}') \quad \text{if} \quad \boldsymbol{t} \neq \boldsymbol{t}'. \tag{A· 5}$$

Actually $\phi(\boldsymbol{t}(J)) = \boldsymbol{t}(J^{\mathrm{a}})$ or $\phi(\boldsymbol{t}(J)) = \boldsymbol{t}(J)$. i.e., when a TEP $\boldsymbol{t}(J)$ is newly generated in the ν-th iteration of the proposed decoding algorithm, then the same iteration of the IBF decoding algorithm possesses the corresponding TEP of the form of either its adjacent pattern $\boldsymbol{t}(J^{\mathrm{a}})$ in the list A or $\boldsymbol{t}(J)$ itself in the list $M^{(\mu(J))}$. Based on this mapping, we can see that there is a correspondence between the TEP newly generated in the proposed decoding algorithm and a TEP stored in list in the IBF decoding algorithm. Therefore, the increased list sizes in the ν-th iteration of both algorithms are the same.

From the arguments (i) and (ii), we can prove the theorem. □

Unfortunately, as known to the authors, there are no heuristic functions which satisfy both conditions (C1) and Eq. (9) except for the function $\Delta(\cdot)$. The function $\Delta(\cdot)$ is ineffective heuristic function compared with the function $f(\cdot)$ or $g(\cdot)$ since it only utilizes the information over the k MRI positions while functions $f(\cdot)$ and $g(\cdot)$ utilize one over the remaining $n - k$ positions as well as one over the k MRI positions. Therefore we may say that the proposed decoding algorithm is more effective than the method in [8].

Hideki Yagi was born in Yokohama, Japan, on Oct. 14, 1975. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 2001 and 2003, respectively. He is currently a doctoral student in Industrial and Management Systems Engineering at Graduate School of Waseda University. Since 2005, he has been a research associate at Media Network Center of Waseda University. His research interests are coding theory and information security. He is a student member of the Society of Information Theory and its Applications.

Toshiyasu Matsushima was born in Tokyo, Japan, on Nov. 26, 1955. He received the B.E. degree, M.E. degree and Dr.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 1978, 1980 and 1991, respectively. From 1980 to 1986, he was with Nippon Electric Corporation, Kanagawa, Japan. From 1986 to 1992, he was a lecturer at Department of Management Information, Yokohama College of Commerce. From 1993, he was an associate professor and since 1996 has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. His research interests are information theory and its application, statistics and artificial intelligence. He is a member of the Society of Information Theory and Its Applications, the Japan Society for Quality Control, the Japan Industrial Management Association, the Japan Society for Artificial Intelligence and IEEE.

Shigeichi Hirasawa was born in Kobe, Japan, on Oct. 2, 1938. He received the B.S. degree in mathematics and the B.E. degree in electrical communication engineering from Waseda University, Tokyo, Japan, in 1961 and 1963, respectively, and the Dr.E. degree in electrical communication engineering from Osaka University, Osaka, Japan, in 1975. From 1963 to 1981, he was with the Mitsubishi Electric Corporation, Hyogo, Japan. Since 1981, he has been a professor of School of Science and Engineering, Waseda University, Tokyo, Japan. In 1979, he was a Visiting Scholar in the Computer Science Department at the University of California, Los Angels (CSD, UCLA), CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. In 2002, he was also a Visiting Faculty at CSD, UCLA. From 1987 to 1989, he was the Chairman of Technical Group on Information Theory of IEICE. He received the 1993 Achievement Award, and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In 1996, he was the President of the Society of Information Theory and Its Applications (Soc. of ITA). His research interests are information theory and its applications, and information processing systems. He is an IEEE Fellow, and a member of Soc. of ITA, the Operations Research Society of Japan, the Information Processing Society of Japan, the Japan Industrial Management Association, and Informs.