The 27th Symposium on Information Theory
and Its Applications (SITA2004)
Gero, Gifu, Japan, Dec. 14–17, 2004

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

Hideki YAGI * Toshiyasu MATSUSHIMA * Shigeichi HIRASAWA *

Abstract— 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. One of the well-known heuristic search methods for MLD is the A* decoding algorithm proposed by Han et al. Based on the decoding algorithms both by Battail and Fang (and its improved technique by Valembois and Fossorier) and by the present authors, we deduce a new method for reducing the space complexity of the A* decoding algorithm. Simulation results show the high efficiency of the proposed method.

Keywords— maximum likelihood decoding, binary block codes, heuristic search, most reliable basis, reliability

1 Introduction

In this paper, we consider the priority-first search-type MLD algorithms where candidate codewords are generated in increasing value of the heuristic function. To the authors' knowledge, G. Battail and J. Fang first proposed a priority-first search method for MLD where a simple evaluation function is employed [1] (we will call this method the BF decoding algorithm). One of the most well-known priority-first search methods for MLD is the A* decoding algorithm proposed by Han et al. [2]. Recently, A. Valembois and M. Fossorier have indicated that a slight modification makes the BF decoding algorithm equivalent to the A* decoding algorithm [6, 7]. Subsequently, Valembois et al. [6] and the present authors [8] have proposed techniques for drastically reducing the space complexity of modified BF decoding algorithms employing some class of heuristic (or evaluation) functions. However, their techniques cannot be straightforwardly used for reducing the space complexity of the A* decoding algorithm in which the search is guided by more sophisticated heuristic functions than that considered in [6, 8].

In this paper, based on the Valembois' indication and the techniques in [8], we propose a method for reducing the space complexity of the A* decoding algorithm. Consequently, we show theoretically that the space complexity for the proposed decoding algorithm is less than that for the A* decoding algorithm, and by computer simulations that the former is significantly reduced.

*Dept. of Industrial and Management Systems Engineering, Waseda University, 3-4-1 Ohkubo Shinjuku-ku, Tokyo, 169-8555 Japan, E-mail: yagi@hirasa.mgmt.waseda.ac.jp

2 Reliability-based MLD Algorithm

Let $\mathcal{C}$ 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 $\mathcal{C}$ by G and the weight profile of $\mathcal{C}$ by $W(\mathcal{C})$. We assume any codewords $\boldsymbol{c} = (c_1, c_2, \ldots, c_n) \in \{0,1\}^n$ of $\mathcal{C}$ are transmitted over the Additive White Gaussian Noise (AWGN) channel. The receiver maps a received sequence $\boldsymbol{r} = (r_1, r_2, \ldots, r_n) \in \mathcal{R}^n$ into a 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, a hard-decision 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 soft-decision decoder estimates the transmitted codeword from $\boldsymbol{\theta}$ and $\boldsymbol{z}$.

In reliability-based decoding algorithms, we use the column-permuted systematic generator matrix $\tilde{G}$ where the leftmost k positions are the *most reliable and linearly independent* (MRI) [2, 5, 6, 7] in non-increasing value of reliability. 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 $\tilde{\mathcal{C}}$ be the code whose codewords are generated by $\tilde{G}$. 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., $u_j = \tilde{z}_j, 1 \leq \forall j \leq k$. 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 and this procedure is repeated until a sufficient condition for the ML codeword is satisfied[1].

Definition 1 For a location set $J \subseteq [1, k]$, the test error pattern (TEP) $\boldsymbol{t}(J) = \big(t_1(J), t_2(J), \ldots, t_k(J)\big)$ has element one in J. Such J is called the *support* of $\boldsymbol{t}(J)$. Define that $b(J)$ be the rightmost position in J, i.e., $b(J) = \max J$. For $j > b(J)$, the TEP $\boldsymbol{t}(J \cup \{j\})$ (or simply $\boldsymbol{t}(J \cup j)$) is called an *extended pattern* of $\boldsymbol{t}(J)$. For $j > b(J)$ and $J' = J \setminus b(J)$, the TEP $\boldsymbol{t}(J' \cup j)$ is called an *adjacent pattern* of $\boldsymbol{t}(J)$ in j. □

For a binary vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_n) \in \{0,1\}^n$, we define the *correlation discrepancy* [6, 7] of $\boldsymbol{v}$ as

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

It is well-known that $\tilde{\boldsymbol{c}}_{\text{best}}$ is the ML codeword if and only if $L(\tilde{\boldsymbol{c}}_{\text{best}}) = \min_{\tilde{\boldsymbol{c}} \in \tilde{\mathcal{C}}} L(\tilde{\boldsymbol{c}})$.

[1] $\oplus$ represents Exclusive OR operation.

3 Priority-first Search Method of Test Error Patterns

The A* decoding algorithm [2] searches the ML codeword through the trellis or the binary tree of the code. Valembois and Fossorier have indicated that the modified BF algorithm and the A* decoding algorithm are equivalent when both algorithms use the same heuristic function. In this section, we state the A* decoding algorithm from the Valembois' perspective [6].

We first address the heuristic function of the search. For $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{\mathcal{C}}$ and $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$, we define

$$T\big(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) = \Big\{\boldsymbol{v}\,\big|\, v_j = u_j \oplus t_j \ \text{ for } j \in [1,k], \ \text{and } d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) \in W(\tilde{\mathcal{C}})\Big\}, \quad (2)$$

where $d_H(\cdot,\cdot)$ denotes the Hamming distance. Then the heuristic function considered in [2] is defined as

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})} \Big\{L(\boldsymbol{v})\Big\}. \quad (3)$$

The A* decoding algorithm generates $\boldsymbol{t}$ in increasing value of $f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})$. Such $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$ is called the *referenced codeword* [3, 6] or the *seed* [2, 5].

The A* decoding algorithm performs the priority-first search with not only the function f but any heuristic functions F satisfying the following condition: for $j \notin J$,

$$\textbf{(C1)} \qquad F\big(\boldsymbol{t}(J)\big) \le F\big(\boldsymbol{t}(J \cup j)\big). \quad (4)$$

It is guaranteed that the A* decoding algorithm finds the most likely codeword if

$$F\big(\boldsymbol{t}(J)\big) \le L(\tilde{\boldsymbol{c}}_J), \quad (5)$$

where $\tilde{\boldsymbol{c}}_J = \tilde{\boldsymbol{c}}_\emptyset \oplus \boldsymbol{t}(J)\tilde{G}$.

Hereafter, we describe how to perform the priority-first search of TEPs using the heuristic function satisfying (C1). Let $M^{(1)}, M^{(2)}, \ldots, M^{(k)}$ be k lists of TEPs. The TEP $\boldsymbol{t}(J)$ is supposed to be in $M^{(b(J))}$ where $b(J) = \max J$. In a list $M^{(j)}, \forall j \in [1,k]$, TEPs are ordered in increasing value of a heuristic function F.

By the condition (C1), the TEPs with the minimum value of F in $M^{(j)}, j \in [1,k]$, is $\boldsymbol{t}(j)$ whose Hamming weight are one. Therefore, we just need to set the initial lists as $M^{(j)} = \big\{\boldsymbol{t}(j)\big\}$ for $j \in [1,k]$. Then, the algorithm searches the TEP with the minimum value of F (we will call this pattern the *best pattern*) among the set of those that have not been found.

We here describe the A* decoding algorithm which is equivalent to the BF algorithm with a slight modification.

[The A* decoding algorithm]

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

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

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 $\tilde{\boldsymbol{c}}_{\mathrm{best}} := \tilde{\boldsymbol{c}}_J$.

S4) For all lists $M^{(j)}$ such that $j > b(J)$, insert the extended patterns $\boldsymbol{t}(J \cup j)$ at the position such that the list remains increasing order. Delete $\boldsymbol{t}(J)$ from $M^{(b(J))}$.

S5) If $M^{(j)} = \emptyset$ for all $j \in [1,k]$, then output $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ and halt the algorithm. Otherwise, go to S2). □

In S4), we need to sort the extended pattern so that the list $M^{(j)}$ remains increasing order of the heuristic values. By sorting, the priority-first search of the A* decoding algorithm is maintained [6].

In the original A* decoding algorithm, the only one list of TEPs is used. If we combine the k lists into the united list and order test patterns increasing order of the heuristic values in it, then the above algorithm becomes identical to the original A* decoding algorithm although the behaviors of the two algorithms seem different.

We here describe the complexity of the A* decoding algorithm. In a decoding procedure of a received sequence $\boldsymbol{r}$, the space complexity is $O(k \times M(\boldsymbol{r}))$ where $M(\boldsymbol{r})$ represents the maximum number of TEPs stored in lists. As for the time complexity, that for generating TEPs is dominant as well as that for encoding them.

4 Proposed Decoding Algorithm

In this section, we propose a method for reducing the list size of TEPs in the A* decoding algorithm which dominates the space complexity.

We here define the following condition (C2) for a heuristic function F.

Definition 2 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 satisfies

$$\textbf{(C2)} \qquad F\big(\boldsymbol{t}(J \cup j_1)\big) \ge F\big(\boldsymbol{t}(J \cup j_2)\big). \quad (6)$$

We will call (6) the condition (C2). □

For the function f, we show the following lemma.

Lemma 1 For a referenced codeword $\tilde{\boldsymbol{c}}_{\mathrm{ref}} \in \tilde{\mathcal{C}}$, let $\boldsymbol{t}_{\mathrm{ref}}$ be the TEP of $\tilde{\boldsymbol{c}}_{\mathrm{ref}}$. Assuming that $S^{(1)}$ and $S^{(0)}$ be the support of $\boldsymbol{t}_{\mathrm{ref}}$ and its complement, respectively. Then for $\boldsymbol{t}(J)$, the heuristic function $f\big(\boldsymbol{t}(J), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big)$ satisfies the condition (C2).

(**Proof**) From (1) and (3), the heuristic function $f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})$ of a TEP $\boldsymbol{t} = (t_1, t_2, \ldots, t_k)$ satisfies

$$f(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}}) = \sum_{j=1}^{k} t_j |\tilde{\theta}_j| + \min_{\boldsymbol{v} \in T(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})} \Big\{ \sum_{j=k+1}^{n} (\tilde{z}_j \oplus v_j)|\tilde{\theta}_j| \Big\}. \quad (7)$$

Denote the second term of the r.h.s. of (7) by $A(\boldsymbol{t}, \tilde{\boldsymbol{c}}_{\mathrm{ref}})$.

Assuming that $j_1 < j_2$, and $j_1, j_2 \in S^{(0)}$, then TEPs $\boldsymbol{t}(J \cup j_1)$ and $\boldsymbol{t}(J \cup j_2)$ have the same Hamming distance from $\boldsymbol{t}_{\mathrm{ref}}$. Therefore by (2), (3) and (7),

$$A\big(\boldsymbol{t}(J \cup j_1), \tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) = A\big(\boldsymbol{t}(J \cup j_2), \tilde{\boldsymbol{c}}_{\mathrm{ref}})\big). \quad (8)$$

Hence we have

$$f\big(\boldsymbol{t}(J\cup j_1),\tilde{\boldsymbol{c}}_{\mathrm{ref}}\big)-f\big(\boldsymbol{t}(J\cup j_2),\tilde{\boldsymbol{c}}_{\mathrm{ref}}\big) = \sum_{j\in J\cup j_1}|\tilde{\theta}_j| - \sum_{j\in J\cup j_2}|\tilde{\theta}_j| = |\tilde{\theta}_{j_1}|-|\tilde{\theta}_{j_2}| \ge 0.$$

This inequality shows the function f satisfies (C2) when $j_1, j_2 \in S^{(0)}$. In the case that $j_1, j_2 \in S^{(1)}$, we can prove the lemma similarly. □

In the following, we consider heuristic functions that satisfy both (C1) and (C2).

The strategy of the proposed method is like *lazy evaluation* where any TEPs are not generated as long as possible. This approach is similar to improved techniques in [6, 8] where other heuristic functions are considered. We first consider k lists $M^{(j)}$ as in the A* decoding algorithm of Sect. 3. Hereafter, we assume $S^{(0)}=\{i_1,i_2,\ldots,i_s\}$ and $S^{(1)}=\{i'_1,i'_2,\ldots,i'_p\}$.

By the condition (C1), the best pattern in a list $M^{(j)}, j\in[1,k]$, is $\boldsymbol{t}(j)$ whose Hamming weight is one. Furthermore, 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 may as well construct the initial lists as

$$M^{(j)} = \begin{cases} \{\boldsymbol{t}(j)\}, & \text{if } j\in\{i_s, i'_p\}; \\ \emptyset, & \text{otherwise.} \end{cases} \tag{9}$$

At S2) of the A* decoding algorithm, if $\boldsymbol{t}(J)\in M^{(b(J))}$ is chosen as the best pattern, $k-b(J)$ 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 $F\big(\boldsymbol{t}(J\cup j)\big)\ge F\big(\boldsymbol{t}(J\cup i_s)\big)$ for $\forall j\in S^{(0)}$ and $F\big(\boldsymbol{t}(J\cup j)\big)\ge F\big(\boldsymbol{t}(J\cup i'_p)\big)$ for $\forall j\in S^{(1)}$.

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. Assume that a TEP $\boldsymbol{t}(J\cup i_q)$ such that $i_q > b(J)$ and $i_q\in S^{(0)}$ has been already stored in the list $M^{(i_q)}$. Since $\boldsymbol{t}(J\cup j)$ such that $j<i_q$ and $j\in S^{(0)}$ cannot be the best pattern, we may as well store these extended patterns only after $\boldsymbol{t}(J\cup i_q)$ is chosen as the best pattern at S2). If $i_{q-1}>b(J)$, $\boldsymbol{t}(J\cup i_{q-1})$ has the smallest heuristic value among all adjacent patterns of $\boldsymbol{t}(J\cup i_q)$ in $S^{(0)}$ from the condition (C2), i.e.,

$$F\big(\boldsymbol{t}(J\cup i_{q-1})\big) = \min_{j\in S^{(0)}}\Big\{F\big((\boldsymbol{t}J\cup j)\big)\;\big|\; b(J)<j<i_q\Big\}. \tag{10}$$

Therefore, after $\boldsymbol{t}(J\cup i_q)$ is chosen as the best pattern at S2), $\boldsymbol{t}(J\cup i_{q-1})$ is inserted into the list $M^{(i_{q-1})}$. This modification reduces the space complexity significantly. We note that the next generated TEP $\boldsymbol{t}(J\cup i_{q-1})$ and $F\big(\boldsymbol{t}(J\cup i_{q-1})\big)$ are easily calculated from the selected pattern $\boldsymbol{t}(J\cup i_q)$ and $F\big(\boldsymbol{t}(J\cup i_q)\big)$. Similar arguments also hold when $\boldsymbol{t}(J\cup i'_q), i'_q\in S^{(1)}$, has been the best pattern at S2).

We describe a proposed decoding algorithm employing the above method.

[The proposed decoding algorithm]

P1) Set $\tilde{\boldsymbol{c}}_\emptyset := \boldsymbol{u}\tilde{G}$, $\tilde{\boldsymbol{c}}_{\mathrm{best}} := \tilde{\boldsymbol{c}}_\emptyset$ and $\underline{L} := L(\tilde{\boldsymbol{c}}_\emptyset)$. Construct the initial lists of TEPs by (9).

P2) Choose the best pattern $\boldsymbol{t}(J)\in M^{(b(J))}$ among nonempty lists. If $F(\boldsymbol{t}(J))\ge \underline{L}$, then output $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ 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 $\tilde{\boldsymbol{c}}_{\mathrm{best}} := \tilde{\boldsymbol{c}}_J$.

P4) a) If $b(J)=i_q$ (i.e., $b(J)\in S^{(0)}$) and the adjacent pattern $\boldsymbol{t}(J'\cup i_{q-1})$ exists where $J' = J\setminus b(J)$, then insert it into the list $M^{(i_{q-1})}$.
b) If $b(J)=i'_q$ (i.e., $b(J)\in S^{(1)}$) and $\boldsymbol{t}(J'\cup i'_{q-1})$ exists where $J' = J\setminus b(J)$, then insert it into $M^{(i'_{q-1})}$.
c) If $b(J)<i_s$, then insert $\boldsymbol{t}(J\cup i_s)$ into $M^{(i_s)}$. If $b(J)<i'_p$, then insert $\boldsymbol{t}(J\cup i'_p)$ into $M^{(i'_p)}$. Delete $\boldsymbol{t}(J)$ from $M^{(b(J))}$.

P5) If $M^{(j)}=\emptyset$ for all $j\in[1,k]$, then output $\tilde{\boldsymbol{c}}_{\mathrm{best}}$ and halt the algorithm. Otherwise, go to S2). □

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

In terms of the time and space complexity of the proposed decoding algorithm, we show the following theorems.

Theorem 1 The proposed decoding algorithm performs MLD. Then, the maximum list size of TEPs in the proposed decoding algorithm is less than that in the A* decoding algorithm, if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2). □

Theorem 2 The number of generated TEPs in the proposed decoding algorithm is no more than that in the A* decoding algorithm, if both decoding algorithms employ the same heuristic function satisfying (C1) and (C2). □

5 Simulation Results

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

5.1 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 A* decoding algorithm (we denote it by "A*" in tables) and the proposed decoding algorithm (we denote it by "Proposed" in tables). At each signal to noise ratio (SNR) E_{b}/S_0 [dB], both decoding algorithms are carried out 10,000 times. 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 $\boldsymbol{r}$
Ave : the average value among 10,000 decoding
Max : the maximum value among 10,000 decoding

We use the function f as the heuristic function in both decoding algorithms. We assume that the weight

Table 1: The results of decoding for (63,30,13) BCH code

E_b/N_0 [dB]			A*	Proposed
5.0	Ave	$N(\boldsymbol{r})$	33.7	1.78
		$M(\boldsymbol{r})$	1.19	0.176
	Max	$M(\boldsymbol{r})$	2064	437
4.0	Ave	$N(\boldsymbol{r})$	93.6	21.9
		$M(\boldsymbol{r})$	11.4	2.19
	Max	$M(\boldsymbol{r})$	9579	3272
3.0	Ave	$N(\boldsymbol{r})$	583	225
		$M(\boldsymbol{r})$	96.6	24.4
	Max	$M(\boldsymbol{r})$	14321	5103
2.0	Ave	$N(\boldsymbol{r})$	2908	1305
		$M(\boldsymbol{r})$	553	156
	Max	$M(\boldsymbol{r})$	70519	28005

Table 2: The results of decoding for $(104, 52, 20)$ QR code

E_b/N_0 [dB]			A*	Proposed
6.0	Ave	$N(\boldsymbol{r})$	48.5	0.239
		$M(\boldsymbol{r})$	0.248	0.0342
	Max	$M(\boldsymbol{r})$	187	32
5.0	Ave	$N(\boldsymbol{r})$	82.4	5.79
		$M(\boldsymbol{r})$	3.60	0.471
	Max	$M(\boldsymbol{r})$	4932	793
4.0	Ave	$N(\boldsymbol{r})$	1237	357
		$M(\boldsymbol{r})$	166	30.1
	Max	$M(\boldsymbol{r})$	462986	98620
3.0	Ave	$N(\boldsymbol{r})$	33806	13010
		$M(\boldsymbol{r})$	5461	1297
	Max	$M(\boldsymbol{r})$	11452864	2680552

profiles $W(\mathcal{C})$ of these two codes are unknown and we use their supersets $W'(\mathcal{C}) = \{0, d, d+1, \cdots, n\}$.

We set the reference codeword as $\tilde{\boldsymbol{c}}_{\rm ref} = \tilde{\boldsymbol{c}}_{\rm best}$ for the calculation of (3). When a temporally best codeword $\tilde{\boldsymbol{c}}_{\rm best}$ is newly obtained, the reference codeword is updated. In this case, though we need to store TEPs corresponding to old referenced codewords in memory in the proposed decoding algorithm[2], we need not store its discrepancy which is the real number, while storing TEPs needs their heuristic values.

5.2 Results and Discussion

We show the results of 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 $\mathrm{Max}\, M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 2/5 of that in the A* decoding algorithm at each SNR. Furthermore, the average value of the maximum list size $\mathrm{Ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm is less than 1/3 of that in the A* decoding algorithm. These results show that the effectiveness of the proposed decoding algorithm. By Table 2, the values $\mathrm{Max}\, M(\boldsymbol{r})$ and $\mathrm{Ave}\, M(\boldsymbol{r})$ in the proposed decoding algorithm are less than 1/4 of those in the A* decoding algorithm. Theses results indicate the proposed method also works well for a longer code.

The number of generated TEPs $N(\boldsymbol{r})$ is one of indices to evaluate time complexity in heuristic search MLD algorithms [2, 5, 6]. By Table 1 and 2, $N(\boldsymbol{r})$ in the proposed decoding algorithm are less than 2/5 of $N(\boldsymbol{r})$ in the A* decoding algorithm even at 3.0 [dB]. These results demonstrate the proposed method reduces the time complexity of the A* decoding algorithm as well as the space complexity.

6 Conclusion and Future Works

In this paper, we propose a new priority-first heuristic search method reducing the space complexity of the A* decoding algorithm via the perspective of [6]. The proposed decoding algorithm is guaranteed to perform MLD since the set of generated candidate codewords is identical to that in the original A* decoding algorithm. The proposed decoding algorithm reduces not only the space complexity but the time one in the A* decoding algorithm.

As future works, we need to develop a method for heuristic search MLD algorithm with powerful heuristic functions such as in [5].

[2]Its space complexity may be fairly small since the number of obtained referenced codewords is small. In our simulation, we observed that the maximum number of updating referenced codeword is only 24 among all simulations.

Acknowledgment

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. Inform. Theory*, vol.39, no.5, pp.1514–1523, Sept. 1993.

[3] D. Gazelle and J. Snyders, "Reliability-based code-search algorithm for maximum-likelihood decoding of block codes," *IEEE Trans. Inform. Theory*, vol.43, no.1, pp.239–249, Jan. 1997.

[4] M.P.C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inform. Theory*, vol.41, no.5, pp.1379–1396, Sept. 1995.

[5] 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.

[6] 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.

[7] 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.

[8] H. Yagi, T. Matsushima, and S. Hirasawa, "A method for reducing space complexity of reliability-based heuristic search maximum likelihood decoding algorithms", *Proc. of the 26th Symposium on Information Theory and its Applications (SITA2003)*, pp.185–188, Hyogo, Japan, Dec. 2003.