PAPER

An Efficient Heuristic Search Method for Maximum Likelihood Decoding of Linear Block Codes Using Dual Codes

Tomotsugu OKADA[†*a)], *Nonmember*, **Manabu KOBAYASHI**[†], *Regular Member*, *and* **Shigeichi HIRASAWA**[†], *Fellow*

SUMMARY Y.S. Han et al. have proposed an efficient maximum likelihood decoding (MLD) algorithm using A* algorithm which is the graph search method. In this paper, we propose a new MLD algorithm for linear block codes. The MLD algorithm proposed in this paper improves that given by Han et al. utilizing codewords of dual codes. This scheme reduces the number of generated codewords in the MLD algorithm. We show that the complexity of the proposed decoding algorithm is reduced compared to that given by Han et al. without increasing the probability of decoding error.

key words: *block codes, soft decision, maximum likelihood decoding, A* algorithm, dual codes*

1. Introduction

Soft decision decoding can pull out more capable of error correction than hard decision decoding. Especially, maximum likelihood decoding (MLD) is the best method in the sense of minimizing the probability of decoding error when all codewords have equal probability to be transmitted. However, since the complexity required for performing MLD is impractically large, many researchers have been studying to develop methods for reducing it [2]–[4], [7].

Y.S. Han et al. have proposed an efficient MLD algorithm using A* algorithm [6] which is the graph search method [3]. Hereafter, we call the decoding algorithm given by Han et al. A* decoding algorithm. At low SNR, for moderate code rates and long code lengths, however, the complexity required for performing MLD is still impractically large.

In this paper, we propose a new MLD algorithm for linear block codes. Our MLD algorithm improves A* decoding algorithm by utilizing codewords of dual codes. This scheme reduces the number of generated codewords in the MLD algorithm. We show that the complexity of the proposed decoding method is reduced compared to A* decoding algorithm maintaining the probability of decoding error of MLD.

This paper is organized as follows. In Sect. 2, we describe the method for MLD. In Sect. 3, A* decoding algorithm is briefly reviewed. In Sect. 4, we propose a new heuristic function. Simulation results are presented in Sect. 5 and concluding remarks are given in Sect. 6.

2. Preliminary

Let C be a binary (n, k, d) block code of length n, the number of information symbols k, and minimum distance d with generator matrix G. A codeword $\boldsymbol{c} = (c_1,c_2,\ldots,c_n)$ of C is transmitted over the additive white Gaussian noise (AWGN) channel. Let $\boldsymbol{r} = (r_1,r_2,\ldots,r_n), r_i \in R$, denote a received vector. We now define the bit log-likelihood ratio of r_i as $\phi_i = \ln \frac{Pr(r_i|0)}{Pr(r_i|1)}$. The soft decision decoder estimates the transmitted codeword from $\boldsymbol{\phi} = (\phi_1,\phi_2,\ldots,\phi_n)$ and a hard decision vector $\boldsymbol{z} = (z_1,z_2,\ldots,z_n)$, where z_i is defined as follows:

$$z_i = \begin{cases} 0, & \phi_i \geq 0, \\ 1, & \phi_i < 0. \end{cases} \tag{1}$$

At first, the soft decision decoder obtains the k most reliable and linearly independent columns (Most Reliable Independent Positions:MRIPs) of generator matrix G from $\boldsymbol{\phi}$. Secondly, the vector $\tilde{\boldsymbol{\phi}}$ is obtained from $\boldsymbol{\phi}$ in such that the k MRIPs are re-ordered in the order of decreasing value and other $n-k$ positions are re-ordered in the same manner behind the k MRIPs. That is, for $1 \leq i < j \leq k$, $|\tilde{\phi}_i| \geq |\tilde{\phi}_j|$, and for $k+1 \leq i' < j' \leq n$, $|\tilde{\phi}_{i'}| \geq |\tilde{\phi}_{j'}|$. The vector $\tilde{\boldsymbol{z}}$ and the code $\tilde{C}$ are obtained by permuting the positions of $\boldsymbol{z}$ and C, respectively corresponding to the above re-order. Let $\tilde{G}$ be a generator matrix of $\tilde{C}$ whose leftmost k columns form the $k \times k$ identity matrix.

For any $\boldsymbol{x} \in \{0,1\}^n$, we define *the reliability loss* of $\boldsymbol{x}$ as

$$L(\boldsymbol{x}) = \sum_{i|\tilde{z}_i \neq x_i} |\tilde{\phi}_i|. \tag{2}$$

Then, the most likely codeword $\tilde{\boldsymbol{c}}^*$ of $\tilde{C}$ satisfies $L(\tilde{\boldsymbol{c}}^*) = \min_{\tilde{\boldsymbol{c}} \in \tilde{C}} L(\tilde{\boldsymbol{c}})$.

Manuscript received March 30, 2001.
Final manuscript received October 15, 2001.

[†]The authors are with the Department of the Industrial and Management Systems Engineering, School of Science and Engineering, Waseda University, Tokyo, 169-8555 Japan.
[*]Presently, with Hewlett-Packard Japan, Ltd.
a) E-mail: okada@hirasa.mgmt.waseda.ac.jp

3. A Review of A* Decoding Algorithm [3]

The A* decoding algorithm searches the most likely codeword through the graph of $\tilde{C}$ using A* algorithm.

The initial codeword $\tilde{\boldsymbol{c}}_0$ is obtained as follows. We define $\tilde{\boldsymbol{u}} = (\tilde{u}_1, \tilde{u}_2, \ldots, \tilde{u}_k) \in \{0,1\}^k$ as the leftmost k symbols of $\tilde{\boldsymbol{z}}$, and $\tilde{\boldsymbol{c}}_0$ can be obtained by encoding $\tilde{\boldsymbol{u}}$ with $\tilde{G}$, i.e. $\tilde{\boldsymbol{c}}_0 = \tilde{\boldsymbol{u}}\tilde{G}$.

We define the depth of the start node S as 0. We should notice that the graph of $\tilde{C}$ is a binary code tree by depth k since the first k positions are the k MRIPs.

We define the minimum cost $g(m)$ from S to a node m of depth t as

$$g(m) = \sum_{i=1}^{t} |\tilde{\phi}_i|(\bar{v}_i \oplus \tilde{z}_i), \tag{3}$$

where $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_t$ are the labels of the path from S to m, and $\oplus$ represents exclusive OR operator.

Let $HW = \{w_i | 0 \le i \le I(\le n)\}$ be the set of all distinct Hamming weights that codewords of C may have, and assume $0 = w_0 < w_1 < \cdots < w_I \le n$. From the linear property of $\tilde{C}$, the Hamming distance $d_H(\cdot,\cdot)$ between any two codewords of $\tilde{C}$ must belong to HW. We now give the seed codeword $\tilde{\boldsymbol{c}}_s$ and let the set $T(m)$ denote as follows:

$$\begin{aligned} T(m) = \{&\boldsymbol{v} | \boldsymbol{v} = (\bar{v}_1, \bar{v}_2, \cdots, \bar{v}_t, v_{t+1}, \cdots, v_n) \\ &\text{and } d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_s) \in HW\}, \end{aligned} \tag{4}$$

where $v_i \in \{0,1\}$ for $i = t+1, t+2, \ldots, n$. Furthermore, we denote the heuristic function $\hat{h}(m)$ as follows.

$$\hat{h}(m) = \min_{\boldsymbol{v} \in T(m)} \left\{ \sum_{i=t+1}^{n} |\tilde{\phi}_i|(v_i \oplus \tilde{z}_i) \right\}. \tag{5}$$

In A* algorithm, we expand the node in the increasing order of the estimate function $\hat{f}(m) = g(m) + \hat{h}(m)$. Note that the heuristic function satisfies the condition to find the optimal path since for any node m, $\hat{h}(m) \le h^*(m)$ holds, where $h^*(m)$ is the actual cost of a minimum cost path from node m to the goal node of depth n. That is, the most likely codeword must be found by using A* algorithm with $\hat{f}(\cdot)$ through the graph of $\tilde{C}$.

The decoding algorithm can be stopped at any time that we obtain the codeword $\boldsymbol{c}_\ell = (c_{\ell 1}, c_{\ell 2}, \cdots, c_{\ell n})$ which satisfies the following criterion.

Criterion 1: If

$$\hat{h}(S) = \sum_{i=1}^{n} |\tilde{\phi}_i|(c_{\ell i} \oplus \tilde{z}_i) = L(\boldsymbol{c}_\ell), \tag{6}$$

then $\boldsymbol{c}_\ell$ is the most likely codeword, where $\hat{h}(S)$ is calculated with respect to the seed $\boldsymbol{c}_\ell$.

We should notice that this criterion is equivalent to the acceptance criterion for the most likely codewords in [8]. □

In A* decoding algorithm, we assume that the list $\boldsymbol{OPEN}$ stores the nodes expanded so far in the increasing order of the estimate function $\hat{f}(\cdot)$. Let $\underline{L}$ denote the minimum reliability loss of the codeword generated so far.

[A* decoding algorithm]

```
1) Generate the initial codeword $\tilde{c}_0 := \tilde{u}\tilde{G}$. We set
   $\tilde{c}^* := \tilde{c}_0$; $\tilde{c}_s := \tilde{c}_0$; $\underline{L} := L(\tilde{c}_0)$; $OPEN = \{S\}$.
2) If $\tilde{c}^*$ satisfies (6), then output the codeword $\tilde{c}^*$ and
   finish.
3) Pop the top node $\mathcal{N}$ in $OPEN$. If the depth $\ell$ of $\mathcal{N}$
   is $n$, then output the codeword $\tilde{c}^*$ and finish.
4) If $\ell < k$, then we expand $\mathcal{N}$ to the successor node
   $\mathcal{N}_0$ whose label $\bar{v}_{\ell+1} = 0$. Calculate $\hat{g}(\mathcal{N}_0)$ and $\hat{h}(\mathcal{N}_0)$
   by (3) and (5), respectively. Furthermore, calculate
   $\hat{f}(\mathcal{N}_0) := \hat{g}(\mathcal{N}_0) + \hat{h}(\mathcal{N}_0)$ and push (insert) $\mathcal{N}_0$ to
   $OPEN$. And, for another successor $\mathcal{N}_1$ whose label
   $\bar{v}_{\ell+1} = 1$, we also calculate $\hat{f}(\mathcal{N}_1)$ and push $\mathcal{N}_1$ to
   $OPEN$, then go to 3).
5) If $\ell = k$, then generate the codeword $\tilde{c}$ by using the
   labels $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_k$ and $\tilde{G}$. If $L(\tilde{c}) \le \underline{L}$, then update
   $\underline{L} := L(\tilde{c})$ and $\tilde{c}^* := \tilde{c}$ and push the goal node $\mathcal{M}$
   (depth $n$) corresponding to $\tilde{c}$ to $OPEN$ and go to
   2), otherwise go to 3). □
```

Thus A* decoding algorithm performs MLD. It is important to notice that the seed $\boldsymbol{c}_s$ does not need to be fixed during the decoding of $\tilde{\boldsymbol{\phi}}$. When the seed is changed, we do not recalculate heuristic functions with respect to the new seed for every node in the list $\boldsymbol{OPEN}$. When the seed is allowed to change, we call this algorithm an adaptive procedure. Even though we implement the adaptive procedure, this procedure guarantees to find the most likely codeword since $\hat{h}(m) \le h^*(m)$ holds for any node m.

We now discuss the time complexity of A* decoding algorithm. To permute the reliability vector requires computational complexity of $O(n \log n)$ and to construct $\tilde{G}$ from G, that of $O(n \times \min\{k^2, (n-k)^2\})$ [2], [3]. Note that these procedures are carried out only once for each decoding. On the contrary to them, step 5) (encoding) and step 4) (calculation a heuristic function) can be processed iteratively depending on the signal to noise ratio of the channel. The time complexity of encoding is $O(n \times k)$ and that of calculation for a heuristic function is $O(n)$. Hence the time complexity of this decoding algorithm is $O(nk \times N(\boldsymbol{\phi}))$, where $N(\boldsymbol{\phi})$ is the number of generated codewords during the decoding of $\boldsymbol{\phi}$. The space complexity of this algorithm is $O(n \times M(\boldsymbol{\phi}))$, where $M(\boldsymbol{\phi})$ is the maximum number of nodes stored in the list $\boldsymbol{OPEN}$ during the decoding of $\boldsymbol{\phi}$.

4. Proposed Heuristic Function

In this section, we propose a new heuristic function using a codeword of the dual code.

The A* algorithm has the important property on the efficiency of the algorithm [6].

Lemma 1 ([6]): Let A_1 and A_2 be A* algorithm whose heuristic functions are $\hat{h}_1(\cdot)$ and $\hat{h}_2(\cdot)$, respectively. For any node m, if $\hat{h}_1(m) \leq \hat{h}_2(m)$, then the number of nodes expanded by A_1 are no less than that by A_2. □

Let $\tilde{C}^{\perp}$ be a dual code of $\tilde{C}$. Any codeword $\tilde{\boldsymbol{c}} \in \tilde{C}$ and any codeword $\tilde{\boldsymbol{c}}^{\perp} \in \tilde{C}^{\perp}$ satisfy $\tilde{\boldsymbol{c}} \cdot \tilde{\boldsymbol{c}}^{\perp} = 0$, where "$\cdot$" represents inner product. By utilizing this property, we can improve a heuristic function. Given a codeword $\tilde{\boldsymbol{c}}^{\perp}$ in $\tilde{C}^{\perp}$, for any node m, we denote the set $T_p(m)$ as

$$\begin{aligned} T_p(m) = \{\boldsymbol{v}|\boldsymbol{v} = (\bar{v}_1, \bar{v}_2, \cdots, \bar{v}_t, v_{t+1}, \cdots, v_n),\\ d_H(\boldsymbol{v}, \tilde{\boldsymbol{c}}_s) \in HW, \text{ and } \boldsymbol{v} \cdot \tilde{\boldsymbol{c}}^{\perp} = 0\}. \end{aligned} \quad (7)$$

$T_p(m) \subseteq T(m)$ holds for any node m since $T_p(m)$ is a more stringent set than $T(m)$. Furthermore, by utilizing $T_p(m)$, we propose a new heuristic function $\hat{h}_p(m)$ as

$$\hat{h}_p(m) = \min_{\boldsymbol{v} \in T_p(m)} \left\{ \sum_{i=t+1}^{n} |\tilde{\phi}_i|(v_i \oplus \tilde{z}_i) \right\}. \quad (8)$$

The decoding algorithm can also be stopped at any time when we obtain the codeword $\boldsymbol{c}_\ell = (c_{\ell 1}, c_{\ell 2}, \cdots, c_{\ell n})$ which satisfies the following criterion.

Criterion 2: If

$$\hat{h}_p(S) = \sum_{i=1}^{n} |\tilde{\phi}_i|(c_{\ell i} \oplus \tilde{z}_i) = L(\boldsymbol{c}_\ell), \quad (9)$$

where $\hat{h}_p(S)$ is calculated with respect to the seed $\boldsymbol{c}_\ell$, then $\boldsymbol{c}_\ell$ is the most likely codeword. □

Hereafter, we call A* decoding algorithm using the proposed heuristic function $\hat{h}_p(m)$ *proposed decoding algorithm.*

We should notice that $\hat{h}(m) \leq \hat{h}_p(m) \leq h^*(m)$ since $T(m) \supseteq T_p(m) \supseteq C(m)$, where $C(m)$ is a set of codewords passing through node m. Hence, in case that we do not use the stopping criterion at the step 2), Lemma 1 and this property lead to the following theorem without proof.

Theorem 1: The proposed decoding algorithm does not expand more nodes than A* decoding algorithm does. Furthermore, The number of encoding during the proposed decoding method are no more than the number of generated codewords during A* decoding algorithm. □

When we use the stopping criterion and we implement the adaptive procedure, we cannot theoretically ensure that the proposed decoding algorithm is superior to the original A* decoding algorithm. However, in these conditions, we may anticipate that the proposed decoding algorithm is more efficient than the original A* decoding algorithm in almost all cases. The reason is that the proposed decoding algorithm tends to visit the most likely codeword faster than the original A* decoding algorithm does.

5. Simulation Results

In this section, we present simulation results for the (104,52,20) binary extended quadratic residue (QR) code and the (128,64,22) binary extended BCH code. We assume that these codes are transmitted over the AWGN channel with antipodal signaling. That is, the ith signal (i = 1,2, $\cdots$, n) of the transmitted codeword $\boldsymbol{c}$ and received vector $\boldsymbol{r}$ are $(-1)^{c_i}\sqrt{E_s}$ and $r_i = (-1)^{c_i}\sqrt{E_s} + e_i$, where E_s is the signal energy per channel bit and $\sigma^2(e_i) = N_0/2$ is the double-sided noise spectral density. The signal to noise ratio (SNR) for the channel is $\gamma = E_s/N_0$ and the SNR per transmitted information bit is $\gamma_b = \gamma \cdot n/k$. For the AWGN channel, we have $\phi_i = \frac{2r_i}{\sigma^2(e_i)}$, so we can substitute $\boldsymbol{r}$ for $\boldsymbol{\phi}$ in our simulation.

For the (128,64) code, HW is $\{x|(x = 0)$ or (x is even and $22 \leq x \leq 106$) or $(x = 128)\}$ [1]. We do not know HW exactly for the (104,52) code, so we use a superset for this. For this code, we know that $d = 20$ and the Hamming weight of any codeword is divisible by 4 [5]. Hence, the superset used is $\{x|(x = 0)$ or (x is divisible by 4 and $20 \leq x \leq 84$) or $(x = 104)\}$.

Firstly, we analyze which codewords in $\tilde{C}^{\perp}$ has the strong influence to make the algorithm efficient. We examined 10000 samples for the (15,7,5)BCH codes in the following way. Given the received vector $\boldsymbol{r}$, for each node, we calculate 2^{n-k} heuristic functions from the all codewords in $\tilde{C}^{\perp}$ and find the codeword in $\tilde{C}^{\perp}$ which makes the value of the heuristic function the closest to the actual cost for that node. For each node, we find such codeword. Table 1 shows 5 codewords which have the greatest influence on the efficiency of the algorithm for each SNR. In this table, the numbers from 0 to 255 stand for the index of codewords in $\tilde{C}^{\perp}$, e.g. 128 represents $\tilde{\boldsymbol{c}}^{\perp}_{128} = (\tilde{c}^{\perp}_{128,1}, \tilde{c}^{\perp}_{128,2}, \cdots, \tilde{c}^{\perp}_{128,7}, 10000000) \in \tilde{C}^{\perp}$ where $\tilde{c}^{\perp}_{128,i} \in \{0,1\}$ (i=1,2,$\cdots$,k) and 8 represents $\tilde{\boldsymbol{c}}^{\perp}_{8} = (\tilde{c}^{\perp}_{8,1}, \tilde{c}^{\perp}_{8,2}, \cdots, \tilde{c}^{\perp}_{8,7}, 00001000) \in \tilde{C}^{\perp}$ where $\tilde{c}^{\perp}_{8,i} \in \{0,1\}$ (i=1,2,$\cdots$,k). These codewords are obtained in the following way. Firstly, a parity check matrix $\tilde{H}$ of the form $[A^T|I_{n-k}]$, where I_{n-k} is the $(n-k)\times(n-k)$ identity matrix and A is obtained from $\tilde{G}$ of the form $[I_k|A]$. Secondly, $\tilde{\boldsymbol{c}}^{\perp}_{128}$ and $\tilde{\boldsymbol{c}}^{\perp}_{8}$ are generated by multiplication of $n-k$ tuples (10000000) and of (00001000)

Table 1 The analysis of the codewords in dual codes for the (15,7) codes.

γ_b[dB]		1	2	3	4	5
4.0	index	128	64	32	1	16
	rate[%]	39.7	18.1	8.7	5.8	4.7
3.0	index	128	64	32	16	1
	rate[%]	41.2	19.2	9.5	5.2	4.5
2.0	index	128	64	32	16	1
	rate[%]	42.7	20.3	10.3	5.7	3.5
1.0	index	128	64	32	16	8
	rate[%]	43.9	21.3	10.9	6.0	3.5
0.0	index	128	64	32	16	8
	rate[%]	45.0	22.1	11.4	6.2	3.6

Table 2 Simulation results for the (128,64) codes.

γ_b[dB]			Original A*	Proposed α	Proposed β
6.00	ave	$C(\boldsymbol{r})$	1.01	9.85×10^{-1}	9.85×10^{-1}
		$N(\boldsymbol{r})$	2.03	1.47	1.44
		$M(\boldsymbol{r})$	8.75×10^{-1}	6.49×10^{-1}	6.40×10^{-1}
	max	$M(\boldsymbol{r})$	1659	868	980
5.50	ave	$C(\boldsymbol{r})$	1.19	1.10	1.10
		$N(\boldsymbol{r})$	6.77	4.77	4.68
		$M(\boldsymbol{r})$	2.25	1.62	1.60
	max	$M(\boldsymbol{r})$	2359	1293	1548
5.00	ave	$C(\boldsymbol{r})$	4.10	2.68	2.70
		$N(\boldsymbol{r})$	7.20×10^{1}	4.30×10^{1}	4.37×10^{1}
		$M(\boldsymbol{r})$	9.29	5.58	6.01
	max	$M(\boldsymbol{r})$	10658	4235	4867
4.50	ave	$C(\boldsymbol{r})$	7.93×10^{1}	4.53×10^{1}	4.51×10^{1}
		$N(\boldsymbol{r})$	1.32×10^{3}	7.69×10^{2}	7.65×10^{2}
		$M(\boldsymbol{r})$	8.47×10^{1}	3.67×10^{1}	3.85×10^{1}
	max	$M(\boldsymbol{r})$	189263	81703	94744

by $\tilde{H}$, respectively. In this table, the rate represents the proportion of the number of the nodes for the most influenct codeword on the efficiency of the algorithm to the number of all nodes in percentage terms. If deferent indeces have the same value of heuristic functions, then in such indeces, we count one index whose binary representation has the less Hamming weights. Table 1 shows that $\boldsymbol{c}_{128}^{\perp}$ may be the best codeword in $\tilde{C}^{\perp}$.

Next, we compare the proposed decoding algorithm with the original A* docoding algorithm [3]. From table 1 in the proposed decoding algorithm, $\tilde{\boldsymbol{c}}_{\alpha}^{\perp} = (\tilde{c}_{\alpha,1}^{\perp}, \tilde{c}_{\alpha,2}^{\perp}, \cdots, \tilde{c}_{\alpha,k}^{\perp}, 100\cdots00)$ and $\tilde{\boldsymbol{c}}_{\beta}^{\perp} = (\tilde{c}_{\beta,1}^{\perp}, \tilde{c}_{\beta,2}^{\perp}, \cdots, \tilde{c}_{\beta,k}^{\perp}, 010\cdots00)$ may be the best codeword and the second codeword in $\tilde{C}^{\perp}$, respectively. So, we simulated two versions α and β of the proposed decoding algorithm. The version α utilized $\tilde{\boldsymbol{c}}_{\alpha}^{\perp}$ as the codeword of the dual code, and the version β utilized $\tilde{\boldsymbol{c}}_{\beta}^{\perp}$ as the codeword of the dual code. Table 2 and Table 3 show the simulation results for the (128,64) code and for the (104,52) code, respectively. These results were obtained by simulating 10000 samples for each SNR. In these simulations, we implemented the adaptive procedure and the seed was updated according to the rule in [3].

Table 3 Simulation results for the (104,52) codes.

γ_b[dB]			Original A*	Proposed α	Proposed β
6.00	ave	$C(\boldsymbol{r})$	9.54×10^{-1}	9.37×10^{-1}	9.34×10^{-1}
		$N(\boldsymbol{r})$	1.08	7.70×10^{-1}	7.70×10^{-1}
		$M(\boldsymbol{r})$	4.74×10^{-1}	3.45×10^{-1}	3.49×10^{-1}
	max	$M(\boldsymbol{r})$	1044	499	610
5.50	ave	$C(\boldsymbol{r})$	1.09	1.02	1.02
		$N(\boldsymbol{r})$	3.50	2.36	2.48
		$M(\boldsymbol{r})$	1.18	8.53×10^{-1}	8.77×10^{-1}
	max	$M(\boldsymbol{r})$	1504	738	990
5.00	ave	$C(\boldsymbol{r})$	1.95	1.52	1.53
		$N(\boldsymbol{r})$	2.21×10^{1}	1.39×10^{1}	1.43×10^{1}
		$M(\boldsymbol{r})$	4.40	2.88	3.04
	max	$M(\boldsymbol{r})$	3012	1749	1919
4.50	ave	$C(\boldsymbol{r})$	1.58×10^{1}	9.52	9.53
		$N(\boldsymbol{r})$	2.30×10^{2}	1.40×10^{2}	1.40×10^{2}
		$M(\boldsymbol{r})$	1.93×10^{1}	9.57	1.05×10^{1}
	max	$M(\boldsymbol{r})$	29112	10246	11934

In these tables, the following notations are used.

$N(\boldsymbol{r})$:	the number of nodes visited during the decoding of $\boldsymbol{r}$.
$C(\boldsymbol{r})$:	the number of codewords generated during the decoding of $\boldsymbol{r}$.
$M(\boldsymbol{r})$:	the maximum number of nodes stored on list $\boldsymbol{OPEN}$ during the decoding of $\boldsymbol{r}$.
max:	the maximum value among 10000 samples.
ave:	the average value among 10000 samples.

Simulation results show that the proposed decoding algorithm reduced the number of generated codewords, visited nodes and stored nodes, especially at the low SNR. Furthermore, the version α is more efficient than the version β.

6. Concluding Remarks

We have proposed a new soft-decision MLD algorithm, which is superior to original A* decoding algorithm [3] as stated in Sect. 3.

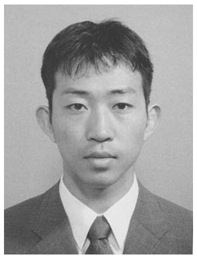
This approach is applicable to any linear block codes. Furthermore, it is able to combine with other decoding method [2], [7]. Our algorithm can easily be converted into various suboptimal procedures.

The theoretical analysis on a method for selecting the codeword of the dual code which is the best for improvement of the efficiency will be remained as a further research.

References

[1] Y. Desaki, T. Fujiwara, and T. Kasami, "The weight distributions of extended binary primitive BCH codes of length 128," IEEE Trans. Inf. Theory, vol.43, no.4, pp.1364–1371, July 1997.

[2] D. Gazelle and J. Snyders, "Reliability-based code-search algorithms for maximum likelihood decoding of block codes," IEEE Trans. Inf. Theory, vol.43, no.1, pp.239–249, Jan. 1997.
[3] Y.S. Han, C.R.P. Hartmann, and C.C. Chen, "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.
[4] M. Kobayashi, T. Matsushima, and S. Hirasawa, "An efficient maximum-likelihood decoding based on reliability information for binary linear block codes," Proc. 22nd Symposium on Information Theory and Its Applications (SITA99), pp.323–326, 1999.
[5] F.J. MacWilliams and N.J.A. Sloane, The Theory of Error-Correcting Codes, Elsevier, New York, 1977.
[6] N.J. Nilsson, Principle of Artifical Intelligence, Tioga Publishing, Palo Alto, 1980.
[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] D.J. Taipale and M.B. Pursley, "An improvement to generalized minimum-distance decoding," IEEE Trans. Inf. Theory, vol.37, no.1, pp.167–172, Jan. 1991.



Tomotsugu Okada was born in Aichi, Japan, on Aug. 25, 1975. He received the B.E. degree and M.E. degree in Industrial and Management Systems Engineering from Waseda University, Tokyo, Japan, in 1999 and 2001, respectively. He is now with Hewlett-Packard Japan, Ltd. His research interest is coding theory.

Manabu Kobayashi was born in Yokohama, Japan, on Oct. 30, 1971. 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 1994, 1996 and 2000, respectively. From 1998 to 2001, He was a research associate in Industrial and Management Systems Engineering at Waseda University, Tokyo, Japan. He is now a reserch fellow in Advanced Research Institute for Science and Engineering Waseda University, Tokyo, Japan. His research intersts are coding theory and data mining. He is a member of the Society of Information Theory and Its Applications 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, 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 Researcher in the Computer Science Department at the University of California, Los Angels, CA. He was a Visiting Researcher at the Hungarian Academy of Science, Hungary, in 1985, and at the University of Trieste, Italy, in 1986. From 1987 to 1989, he was the Chairman of Technical Group on Information Theory of IEICE. He was recieved the 1993 Achievement Award, and the 1993 Kobayashi-Memorial Achievement Award from IEICE. In1996, 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, an IEICE 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.