LC-013

データ圧縮・暗号化及び誤り制御符号化機能を有する統合符号化法

Joint Coding with Data Compression, Encryption, and Error Control Coding Capabilities

金子晴彦* 藤原英二*
Haruhiko Kaneko Eiji Fujiwara

1 はじめに

近年の情報通信システムや計算機システムにおいては，効率的に情報を伝送・蓄積するためのデータ圧縮技術，盗聴や改竄を排除して安全にデータを伝送するための暗号化技術，及び誤りのない高信頼な情報伝送・蓄積を行なうための誤り制御符号化技術が重要な基盤技術となっている [1]．データ圧縮，暗号化及び誤り制御符号化技術の高度化に伴い，情報の符号化・復号装置における計算量は増加する傾向にある．一方，携帯電話等の小型携帯端末，センサネットワークにおけるセンサノード，宇宙機，等においては回路規模や消費電力に制約があることから，計算量の少ない符号化方式が求められる．また，計算機システムやネットワークルータ等においては，高スループットで情報の符号化及び復号を行なう必要があるため，計算量を抑えた符号化方式が必要である．従来，線形符号を用いた計算量の少ない公開鍵暗号 [2] や，データ圧縮法 [3] が提案さているが，データ圧縮，暗号化及び誤り制御符号化の 3 機能をすべて有する符号化法は提案されていない．そこで，本稿ではデータの圧縮，暗号化及び誤り制御符号化が一元的に少ない演算量で実行できる統合符号化法を提案し，本手法の計算機システムへの適用を示す．

2 統合符号化法

2.1 符号化アルゴリズム

Hamming 重み t 以下を有する長さ N_S ビットの情報語を $\mathbf{m} = (m_0, \ldots, m_{N_S-1})^T$ とおく．本稿で提案する統合符号化は，列ベクトルとして表現される情報語 $\mathbf{m}$ に対して，図 1 に示す一連の線形変換を施すことにより，データ圧縮，暗号化及び誤り制御符号化を行なって，長さ N_C ビットを有する符号語 $\mathbf{c} = (c_0, \ldots, c_{N_C-1})^T$ を生成する．線形変換に用いる行列 $\mathbf{H}_S, \mathbf{G}_C, \mathbf{T}, \mathbf{Q}$ 及び $\mathbf{D}$ の定義を以下に示す．ただし，以下では行列及びベクトルはすべてガロア体 GF(2) 上で表現する．

検査行列 $\mathbf{H}_S$: 符号長 N_S ビットを有するランダム t ビット誤り訂正符号 C_S の検査行列であり，M_S 行 N_S 列を有する．検査行列 $\mathbf{H}_S$ は，Hamming 重み t を有する長さ N_S ビットのベクトル $\mathbf{w} = (w_0, \ldots, w_{N_S-1})^T$ を，長さ $M_S \leq N_S$ ビットを有するベクトル

$$\mathbf{v} = (v_0, \ldots, v_{M_S-1})^T = \mathbf{H}_S\mathbf{w}$$

へ圧縮する機能を有する．ここで，$\mathbf{H}_S$ は t ビット誤り訂正符号の検査行列であるから，$\mathbf{w}$ の Hamming 重みが t 以下であれば，$\mathbf{v}$ と $\mathbf{w}$ は一対一で対応する．よって，符号 C_S においてシンドロームによる限界距離復号が可能であれば，$\mathbf{v}$ をシンドロームとして符号 C_S の復号法により誤りパターンを生成することにより，圧縮前のベクトル $\mathbf{w}$ が得られる．符号 C_S としては，低密度パリティ検査 (LDPC) 符号，Goppa 符号，BCH 符号，等を用いることができる．例えば，ベクトル $\mathbf{w}$ が長さ $N_S = 1023$ ビット，Hamming 重み $t = 20$ 以下を有する場合，$\mathbf{H}_S$ として (1023,823,41)BCH 符号の検査行列を用いると，長さ $M_S = 200$ ビットを有するベクトル $\mathbf{v}$ を生成できる．

生成行列 $\mathbf{G}_C$: 通信路誤りを訂正するための，(N_C, M_S) 線形符号 C_C の生成行列であり，M_S 行 N_C 列を有する．生成行列 $\mathbf{G}_C$ は，長さ M_S ビットを有するベクトル $\mathbf{u}_d = (u'_0, \ldots, u'_{M_S-1})^T$ を C_C により符号化し，符号語

$$\mathbf{u} = (u_0, \ldots, u_{N_C-1})^T = \mathbf{G}_C^T\mathbf{u}_d$$

を生成する機能を有する．符号 C_C としては任意の線形ブロック符号を用いることができ，例えば，LDPC 符号，Reed-Solomon(RS) 符号，スポッティバイト誤り制御符号 [4] 等を用いることができる．

置換行列 $\mathbf{T}, \mathbf{Q}$ 及び正則行列 $\mathbf{D}$: 情報語 $\mathbf{m}$ を暗号化するための行列である．ただし，$\mathbf{T}$ は $N_S \times N_S$ 置換行列，$\mathbf{Q}$ は $N_C \times N_C$ 置換行列，$\mathbf{D}$ は $M_S \times M_S$ 正則行列であり，それぞれランダムに構成する．

図 1 より，情報語 $\mathbf{m}$ と符号語 $\mathbf{c}$ は以下の関係を有する．

$$\mathbf{c} = (\mathbf{Q}^{-1}\mathbf{G}_C^T\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1})\mathbf{m}$$

ここで，次式で定義する符号化行列 $\mathbf{A}$ を公開鍵として予め生成しておく．

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{G}_C^T\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1} \quad (1)$$

行列 $\mathbf{A}$ の構成を図 2 にまとめる．行列 $\mathbf{A}$ 以外のすべての行列は秘密鍵とする．以上より統合符号化アルゴリズムは単純に次式で定義できる．

$$\mathbf{c} = \mathbf{A}\mathbf{m}$$

ここで，$\mathbf{A}$ は GF(2) 上の行列であるから，符号化回路は排他的論理和 (EX-OR) ゲートを用いて容易に構成できる．

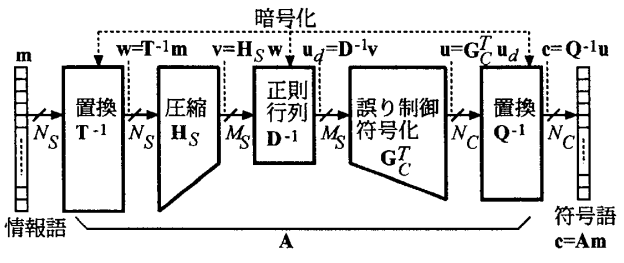


図 1 統合符号化アルゴリズム

2.2 復号アルゴリズム

受信語を $\mathbf{c}' = (c_0, \ldots, c_{N_S-1})^T = \mathbf{c} + \mathbf{n}$ とする．ただし，$\mathbf{c}$ は符号語，$\mathbf{n}$ は長さ N_S を有する誤りベクトルである．秘密鍵を有する正当な受信者が受信語 $\mathbf{c}'$ から情報語 $\mathbf{m}$ を求める手順を示す．受信語 $\mathbf{c}'$ は以下の関係

*東京工業大学 大学院情報理工学研究科

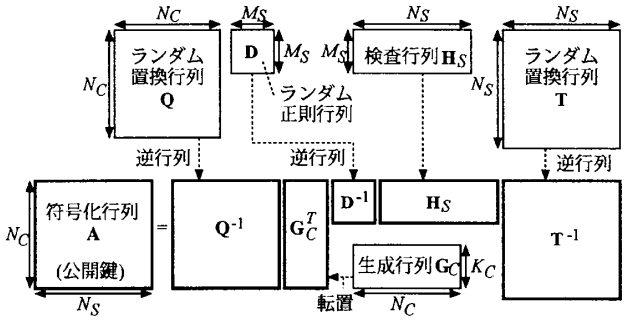


図 2　符号化行列 $\mathbf{A}$ の構成

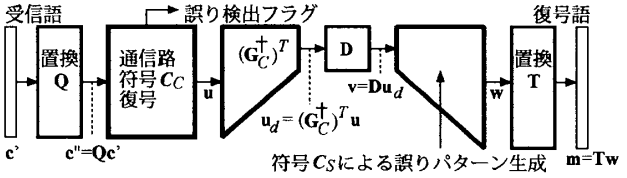


図 3　統合符号化における復号手順

を満足する．

$$\mathbf{c}' = \mathbf{c} + \mathbf{n} = \mathbf{Q}^{-1}\mathbf{G}_C^T\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1}\mathbf{m} + \mathbf{n}$$

受信語 $\mathbf{c}'$ を行列 $\mathbf{Q}$ により行置換する．

$$\mathbf{c}'' = \mathbf{Q}\mathbf{c}' = \mathbf{G}_C^T(\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1}\mathbf{m}) + \mathbf{Q}\mathbf{n}$$

次に，ベクトル $\mathbf{c}''$ を受信語として符号 C_C の復号を行なう．ここで，$\mathbf{G}_C^T(\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1}\mathbf{m})$ は符号 C_C の符号語であり，$\mathbf{Q}\mathbf{n}$ は誤りベクトルを置換したものであるから，$\mathbf{n}$ が訂正可能な誤りであれば，復号により誤りベクトル $\mathbf{Q}\mathbf{n}$ を除去することができる．一方，復号において誤りを検出した場合は，誤り検出フラグを出力して復号を終了する．符号 C_C による復号結果を

$$\mathbf{u} = \mathbf{G}_C^T\mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1}\mathbf{m}$$

とおく．ベクトル $\mathbf{u}$ を以下の式により変換する．

$$\mathbf{u}_d = (\mathbf{G}_C^\dagger)^T\mathbf{u} = \mathbf{D}^{-1}\mathbf{H}_S\mathbf{T}^{-1}\mathbf{m}$$

ただし，$\mathbf{G}_C^\dagger$ は，$\mathbf{G}_C\mathbf{G}_C^\dagger = \mathbf{I}_{M_S}$ を満足する $N_C \times M_S$ 行列，$\mathbf{I}_{M_S}$ は $M_S \times M_S$ 単位行列である．行列 $\mathbf{G}_C^\dagger$ は文献 [5] に示す手法で生成することができる．正則行列 $\mathbf{D}$ と $\mathbf{u}_d$ の積をとり次式を得る．

$$\mathbf{v} = \mathbf{D}\mathbf{u}_d = \mathbf{H}_S\mathbf{T}^{-1}\mathbf{m} = \mathbf{H}_S\mathbf{w}$$

ただし，$\mathbf{w} = \mathbf{T}^{-1}\mathbf{m}$ である．ここで，$\mathbf{T}^{-1}$ は置換行列であるから，$\mathbf{w}$ の Hamming 重みは t 以下である．よって，$\mathbf{v}$ をシンドロームとして符号 C_S による誤りパターン生成を行なうことにより $\mathbf{w}$ が得られる．最後に次式により情報語 $\mathbf{m}$ を得る．

$$\mathbf{T}\mathbf{w} = \mathbf{T}\mathbf{T}^{-1}\mathbf{m} = \mathbf{m}$$

上記の復号手順を図 3 に示す．なお，符号 C_C 及び C_S として LDPC 符号を用い，行列 $\mathbf{D}$ として低密度な正則行列を用いた場合，単一の Tanner グラフを用いた効率的な復号が可能である．

2.3　公開鍵暗号の安全性

提案した統合符号化法が有する公開鍵暗号化機能の安全性について簡単に述べる．

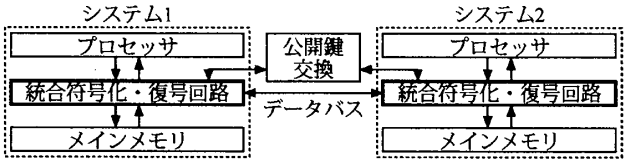


図 4　統合符号化法の計算機システムへの適用

公開鍵に対する攻撃: 公開鍵 $\mathbf{A}$ は秘密鍵である行列 $\mathbf{Q}^{-1}$, $\mathbf{G}_C^T, \mathbf{D}^{-1}, \mathbf{H}_S$ 及び $\mathbf{T}^{-1}$ の積として定義される．ここで，$\mathbf{Q}, \mathbf{D}$ 及び $\mathbf{T}$ はランダムに構成された行列であるから，$\mathbf{A}$ を因数分解して秘密鍵を推定するのは困難である．

暗号文に対する攻撃: 上記の統合符号化法では，式 $\mathbf{c} = \mathbf{A}\mathbf{m}$ の関係により長さ N_S の情報語 $\mathbf{m}$ から符号語 $\mathbf{c}$ を生成する．ここで，行列 $\mathbf{A}$ の階数は $\mathrm{rank}(\mathbf{A}) \leq M_S < N_S$ であるから，逆行列や連立方程式を用いて $\mathbf{c}$ から $\mathbf{m}$ を求めることはできない．よって，$\mathbf{c}$ と $\mathbf{A}$ から $\mathbf{m}$ を求める問題は，ランダムな検査行列で定義される符号の復号問題と等価である．この復号問題は NP 完全問題であると推定されること [6] から符号語 $\mathbf{c}$ に対する効率的な攻撃法は一般に存在しない．ただし，暗号文への反復復号攻撃 [7] に対する安全性は今後評価を行なう必要がある．

暗号の脆弱性:　提案した統合符号化法の暗号化機能に関する脆弱性を以下に示す．

- 本符号は非展性 [1] を有さない．すなわち暗号文 $\mathbf{c} = \mathbf{A}\mathbf{m}$ を解読することなく，$\tilde{\mathbf{c}} = \mathbf{c} + \mathbf{a}_i$ によりベクトル $\mathbf{m}$ の第 i ビットを反転できる．ただし，$\tilde{\mathbf{c}}$ は改ざんされた暗号文であり，$\mathbf{a}_i$ は符号化行列 $\mathbf{A} = [\mathbf{a}_0 \ldots \mathbf{a}_{N_S-1}]$ の第 i 列である．
- 本符号は選択暗号文攻撃に対して脆弱である．すなわち，攻撃対象の暗号文を $\mathbf{c}_t$ とすると，暗号文 $\mathbf{c}_s = \mathbf{c}_t + \mathbf{a}_i$ を選択し，$\mathbf{c}_s$ に対する平文 $\mathbf{m}_s$ を得る．ただし，$\mathbf{a}_i$ は符号化行列 $\mathbf{A}$ の第 i 列である．得られた平文 $\mathbf{m}_s$ の第 i ビットを反転することにより $\mathbf{c}_t$ に対する平文 $\mathbf{m}_t$ が得られる．

上記のような脆弱性は，情報語 $\mathbf{m}$ に対し前処理を施すことにより回避できる [6]．

3　計算機システムへの適用

本節では，統合符号化法を図 4 に示すような計算機システムへ適用する方法を示す．本システムでは，メインメモリ及びデータバスに統合符号化を適用する．データバスの符号化のためにはシステム 1,2 間で公開鍵を交換する必要がある．プロセッサからの出力データを長さ N_K ビットを有する行ベクトル $\mathbf{d} = (d_0, \ldots, d_{N_K-1})$ で表す．ベクトル $\mathbf{d}$ を長さ b ビットを有する $n_k = \frac{N_K}{b}$ 個のバイトに分割したとき，この中で非零であるバイトの数をバイト重みと称し $s = w_b(\mathbf{d})$ で表す．ただし，N_K は b の倍数である．符号構成においては以下の点を考慮する．

データの特性:　プロセッサからの出力データの例を図 5 に示す．ただし，値は 16 進表記である．図 5(a) に示すように，出力データにおいてはビット値"1"が長さ $b = 8$ ビットを有するバイト内に集中して出現する場合があることから，統合符号化においてもバイトを考慮し

```
(a) 6C 00 6C 00 00 00 00 00 14 00 00 00 0A 00 00 00
    00 00 00 00 00 00 00 00 00 00 00 00 03 00 00 00
    F4 02 00 00 02 00 00 00 0C 03 00 00 02 00 00 00
    14 03 00 00 04 00 00 00 1C 03 00 00 2C 00 00 00
    A4 00 00 00 5C 00 00 00 BC 00 00 00 44 00 00 00
    74 00 00 00 8C 00 00 00 D4 00 00 00 47 73 48 64
    28 00 00 00 01 00 00 00 01 00 00 00 01 00 00 00
    1A 00 00 00 48 01 00 00 00 00 00 00 28 00 00 00
(b) 95 03 B2 E8 B3 C2 22 0F 3D 58 E6 B7 29 B6 BA 9D
    A3 02 77 8C 7F 4E D2 C9 85 2E 4B 0E A5 0A A4 A9
    41 A0 49 93 E3 84 1F 70 68 3C DC 80 C9 2B 71 92
    C1 DA 9A A1 5E 38 83 9E 88 C4 23 12 9F C7 DF F3
    F7 05 5E BC 9B 8F 36 7E 0D BF 91 97 4D 63 AF AD
    8D 6F 90 E2 17 F9 69 31 C0 25 B4 ED 8B C5 E7 55
    AB 39 1E 94 98 6C 43 11 26 DE EE 66 87 BB 56 44
    D3 CD E5 CE AA F0 2C 62 C3 6A C6 CF 19 AE F2 61
```

図 5　メインメモリ内のデータの例

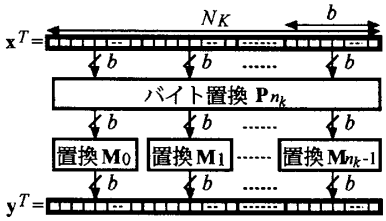
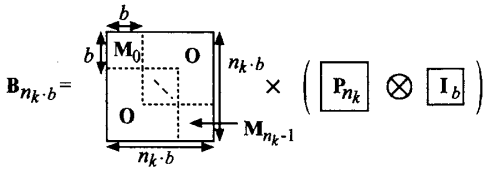


図 6　行列 $\mathbf{B}_{n_k \cdot b}$ による行置換 $\mathbf{y} = \mathbf{B}_{n_k \cdot b}\mathbf{x}$

た処理を行うことにより，効率のよいデータ圧縮が行える．よって，検査行列 $\mathbf{H}_S$ として GF(2^b) 上の RS 符号を用いるとともに，行列 $\mathbf{T}$ はバイトの境界を考慮して以下に示す置換行列 $\mathbf{B}_{n_k \cdot b}$ を用いて構成する．



ただし，$N_K = n_k b$，$\mathbf{M}_i (i \in \{0, \ldots, n_k - 1\})$ は $b \times b$ 置換行列，$\mathbf{P}_{n_k}$ は $n_k \times n_k$ 置換行列であり，$\otimes$ はクロネッカー積を示す．長さ N_K ビットを有する列ベクトル $\mathbf{x}$ に対する行置換 $\mathbf{y} = \mathbf{B}_{n_k \cdot b}\mathbf{x}$ を図 6 に示す．

暗号の安全性:　検査行列 $\mathbf{H}_S$ として前述の RS 符号をそのまま用いると $\mathbf{H}_S$ の構成にランダム性が失われ，暗号の安全性が損なわれる．そこで，$\mathbf{H}_S$ は RS 符号と短縮化 t ビット誤り訂正符号を組み合わせて構成する．さらに，**2.3** に示す暗号の脆弱性を回避するため，ランダムベクトル $\mathbf{r}$，置換及びハッシュ関数を用いてプロセッサ出力データ $\mathbf{d}$ に対する前処理を行う．ただし，$\mathbf{r}$ は Hamming 重み t を有する長さ N_R のベクトルである．

誤り特性:　計算機のメインメモリがバイト構成を有することを考慮し，符号 C_C としてバイト長 b' を有するスポッティバイト誤り制御符号 [4] 等を用いる．また，行列 $\mathbf{Q}$ として置換行列 $\mathbf{B}_{n_c \cdot b'}$ を用いる．

3.1　符号化行列 A の構成

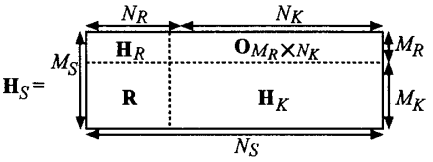
符号化行列 $\mathbf{A}$ は以下の行列を用いて構成する．

置換行列 T:　次式で定義する行列とする．

$$\mathbf{T} = \begin{bmatrix} \mathbf{P}_{N_R} & \mathbf{O}_{N_R \times N_K} \\ \mathbf{O}_{N_K \times N_R} & \mathbf{B}_{n_k \cdot b} \end{bmatrix}$$

ただし，$\mathbf{P}_{N_R}$ は $N_R \times N_R$ 置換行列，$\mathbf{O}_{p \times q}$ は $p \times q$ 零行列である

検査行列 $\mathbf{H}_S$:　次式で定義する行列とする．



ただし，$\mathbf{R}$ は $M_K \times N_R$ ランダム行列，$M_S = M_R + M_K, N_S = N_R + N_K$ であり，部分行列 $\mathbf{H}_R$ 及び $\mathbf{H}_K$ は以下のように定義する．

$\mathbf{H}_R$: 符号長 $N'_R > N_R$ を有する t ビット誤り訂正符号の検査行列から，ランダムに N_R 個の列ベクトルを選択することにより構成した短縮化 t ビット誤り訂正符号 $C_{t\mathrm{EC}}$ の検査行列である．

$\mathbf{H}_K$: 拡大体 GF(2^b) 上の s シンボル誤り訂正 RS 符号 C_{RS} の検査行列を GF(2) 上で表記したものである．ただし，$n_k = \frac{N_K}{b} \leq 2^b - 1$ であり，$n_k < 2^b - 1$ である場合は短縮化 RS 符号とする．また，$s \geq \frac{n_k}{2}$ である場合，$\mathbf{H}_K$ は $N_K \times N_K$ 正則行列とする．

生成行列 $\mathbf{G}_C$:　メインメモリ及びデータバス等における誤り特性に合わせ選択した誤り制御符号の $M_S \times N_C$ 生成行列である．バイト長 b' を有するバイト誤り制御符号，スポッティバイト誤り制御符号，等を用いる．

行列 D 及び Q:　行列 $\mathbf{D}$ を $M_S \times M_S$ 正則行列とする．行列 $\mathbf{Q}$ を $\mathbf{Q} = \mathbf{B}_{n_c \cdot b'}$ により定義する $N_C \times N_C$ 行列とする．ただし，$N_C = n_c b'$ である．

以上の行列を用いて $\mathbf{A}$ を式 (1) により定義する．プロセッサからの出力データ $\mathbf{d}$ は任意のバイト重み $s = w_b(\mathbf{d}) \in \{0, \ldots, n_k\}$ を有することから，統合符号化における符号化行列 $\mathbf{A}$ もバイト重み s に合わせて構成する必要がある．以下では，バイト重み s に対する符号化行列を $\mathbf{A}(s)$ と表す．圧縮率の観点から，すべての $s \in \{0, \ldots, n_k\}$ に対して $\mathbf{A}(s)$ を定めておくことが理想的であるが，実装上困難な場合は代表値 $\{s'_0, s'_1, \ldots, s'_i, \ldots\} \subset \{0, \ldots, n_k\}$ に対し $\mathbf{A}(s'_i)$ を定めておき，出力データ $\mathbf{d}$ は $s = w_b(\mathbf{d}) \leq s'_i$ を満たすような $\mathbf{A}(s'_i)$ を用いて符号化を行なうこともできる．

3.2　符号化アルゴリズム

プロセッサの出力データを $\mathbf{d}$ とし，$\mathbf{d}$ のバイト重みを $s = w_b(\mathbf{d})$ とする．符号化手順を以下に示す．このうち (1) から (4) は暗号の脆弱性回避のための前処理である．

(1) Hamming 重み t を有する長さ N_R のランダムベクトル $\mathbf{r}$ を生成する．

(2) 出力データ $\mathbf{d}$ を置換 $\Pi^{\mathbf{r}}_A$ により置換し，ベクトル $\tilde{\mathbf{d}}$ を生成する．ただし，置換パターンは $\mathbf{r}$ によって決定する．また，$\mathbf{d}$ において同一のバイトに含まれるビットは，$\tilde{\mathbf{d}}$ においても同一のバイトに含まれるように置換 $\Pi^{\mathbf{r}}_A$ を定義する．

(3) ベクトル $\tilde{\mathbf{d}}$ に対するハッシュ値 $\hat{\mathbf{d}} = h(\tilde{\mathbf{d}})$ を求める．

(4) ランダムベクトル $\mathbf{r}$ を置換 $\Pi^{\hat{\mathbf{d}}}_B$ により置換し，ベクトル $\tilde{\mathbf{r}}$ を生成する．ただし，置換パターンは $\hat{\mathbf{d}}$ により決定する．

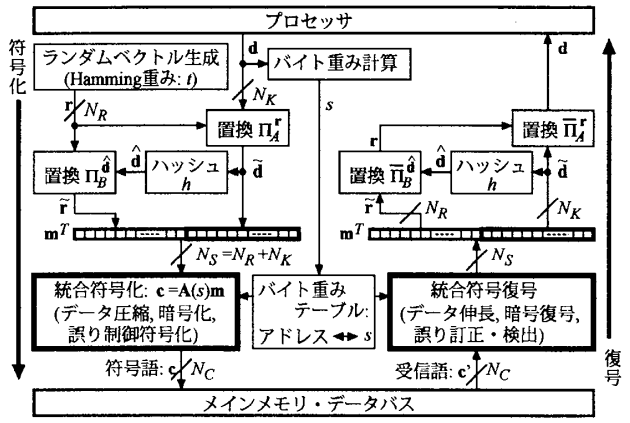


図 7　統合符号化法の計算機システムへの適用

(5) 符号語 $\mathbf{c} = \mathbf{A}(s)\mathbf{m} = \mathbf{A}(s)(\tilde{\mathbf{r}}, \tilde{\mathbf{d}})^T$ を生成する．

上記の符号化アルゴリズムを図 7 中の左に示す．ここで，バイト重みテーブルは各メモリアドレスにおけるデータのバイト重みを保持する．なお，このための制御回路やアドレス変換機構 [8] 等が必要であるが，図 7 では省略している．他のプロセッサへデータを送信するときは，送信先の公開鍵を用いて符号化を行なう．

3.3　復号アルゴリズム

受信語 $\mathbf{c}' = \mathbf{c} + \mathbf{n}$ からベクトル $\mathbf{d}$ を求めるための復号手順は以下のとおりである．

(1) 受信語 $\mathbf{c}'$ を **2.2** に示す手順により復号する．ただし，

$$\mathbf{v} = \mathbf{H}_S\mathbf{T}^{-1}\mathbf{m} = \mathbf{H}_S\mathbf{T}^{-1}(\tilde{\mathbf{r}}, \tilde{\mathbf{d}})^T$$

から $\tilde{\mathbf{r}}$ と $\tilde{\mathbf{d}}$ を求める手順は以下のとおりである．ベクトル $\mathbf{v}$ の先頭 M_R ビットからなるベクトルを $\mathbf{v}_R$ とし，それ以外の M_K ビットからなるベクトルを $\mathbf{v}_K$ とおく．ベクトル $\mathbf{v}_R$ は次の関係を満足する．

$$\mathbf{v}_R = \mathbf{H}_R\mathbf{P}_{N_R}^{-1}\tilde{\mathbf{r}}^T = \mathbf{H}_R\mathbf{r}'$$

ただし，$\mathbf{r}' = \mathbf{P}_{N_R}^{-1}\tilde{\mathbf{r}}^T$ である．ここで，$\mathbf{H}_R$ はランダム t ビット誤り訂正符号 $\boldsymbol{C}_{t\mathrm{EC}}$ の検査行列であり，$\mathbf{r}'$ の Hamming 重みは t であることから，$\mathbf{v}_R$ をシンドロームとして符号 $\boldsymbol{C}_{t\mathrm{EC}}$ による誤りパターン生成を行なうことにより $\mathbf{r}'$ が得られる．一方，$\mathbf{v}_K$ は次の関係を満足する．

$$\mathbf{v}_K = \mathbf{R}\mathbf{P}_{N_R}^{-1}\tilde{\mathbf{r}}^T + \mathbf{H}_K\mathbf{B}_{n_k\cdot b}^{-1}\tilde{\mathbf{d}}^T$$

よって，次式が成立する．

$$\mathbf{v}_K + \mathbf{R}\mathbf{r}' = \mathbf{H}_K\mathbf{B}_{n_k\cdot b}^{-1}\tilde{\mathbf{d}}^T = \mathbf{H}_K\mathbf{d}'$$

ただし，$\mathbf{d}' = \mathbf{B}_{n_k\cdot b}^{-1}\tilde{\mathbf{d}}^T$ である．ここで，左辺の値は既知であり，$\mathbf{H}_K$ は $\mathrm{GF}(2^b)$ 上の s シンボル誤り訂正 RS 符号 $\boldsymbol{C}_{\mathrm{RS}}$ の検査行列であり，かつベクトル $\mathbf{d}'$ のバイト重みは s であるから，$\mathbf{v}_K + \mathbf{R}\mathbf{r}'$ をシンドロームとして符号 $\boldsymbol{C}_{\mathrm{RS}}$ による誤りパターン生成を行なうことにより，$\mathbf{d}'$ が得られる．以上より，$\tilde{\mathbf{d}}^T = \mathbf{B}_{n_k\cdot b}\mathbf{d}'$ 及び $\tilde{\mathbf{r}}^T = \mathbf{P}_{N_R}\mathbf{r}'$ を得る．

(2) ベクトル $\tilde{\mathbf{d}}$ に対するハッシュ値 $\hat{\mathbf{d}} = h(\tilde{\mathbf{d}})$ を求める．

(3) ベクトル $\tilde{\mathbf{r}}$ を $\overline{\Pi}_B^{\hat{\mathbf{d}}}$ により置換しベクトル $\mathbf{r}$ を生成する．ただし，$\overline{\Pi}_B^{\hat{\mathbf{d}}}$ は $\Pi_B^{\hat{\mathbf{d}}}$ の逆置換である．

(4) ベクトル $\tilde{\mathbf{d}}$ を $\overline{\Pi}_A^{\mathbf{r}}$ により置換しベクトル $\mathbf{d}$ を生成する．ただし，$\overline{\Pi}_A^{\mathbf{r}}$ は $\Pi_A^{\mathbf{r}}$ の逆置換である．

上記の復号アルゴリズムを図 7 中の右に示す．

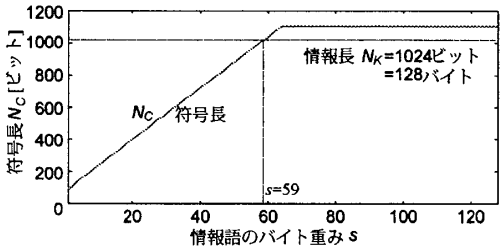


図 8　情報語のバイト重み s と符号長 N_C の関係

4　評価

提案した統合符号化法のデータ圧縮機能の評価として，情報語のバイト重み s と符号長 N_C の関係を図 8 に示す．例として，$N_K = 1024, N_R = 450, b = 8, t = 8, b' = 1$ とし，$\boldsymbol{C}_C$ は短縮化 SEC-DED 符号とする．提案した統合符号化法は，$s < 59$ であるとき情報長 $N_K = 1024$ よりも短い符号長 N_C を有する．圧縮率向上のためには，メインメモリにおけるデータ圧縮法として近年用いられている高速なパターンマッチング [8] 等を基に，バイト重み s を小さくするための前処理を行なう必要がある．

5　まとめ

本稿では，データ圧縮，暗号化及び誤り制御符号化の 3 機能を有する統合符号化法を提案した．線形ブロック符号の生成行列，検査行列及び 3 種の正則行列を基に構成した符号化行列 $\mathbf{A}$ を用いて，情報語 $\mathbf{m}$ から符号語 $\mathbf{c} = \mathbf{A}\mathbf{m}$ を生成する符号化法を示した．また，本稿では提案手法の計算機システムへの適用を目的として，プロセッサからの出力データの特性，暗号の安全性及びメインメモリ等における誤り特性を考慮した符号化行列 $\mathbf{A}$ の構成法を示した．今後は，圧縮率の改善，符号化・復号回路量やスループットの定量的評価，暗号の安全性解析，等を行なう必要がある．

参考文献

[1] 今井秀樹: 情報・符号・暗号の理論; コロナ社.

[2] R. J. McEliece, "A Public-Key Cryptosystem Based on Algebraic Coding Theory," *The Deep Space Network Progress Report*, DSN PR 42–44, pp.114–116, Jan. and Feb. 1978.

[3] D. J. C. MacKay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. Information Theory*, Vol.45, No.2, pp.399–431, March 1999.

[4] E. Fujiwara, *Code Design for Dependable Systems*, Wiley, 2006.

[5] E. Fujiwara, et al., "Parallel Decoding for Burst Error Control Codes," *Electronics and Communications in Japan*, Part III, Vol.87, No.1, pp.38–48, Jan. 2004.

[6] K. Kobara and H. Imai, "Semantically Secure McEliece Public-Key Cryptosystem," *IEICE Trans. Fundamentals*, Vol.E85-A, No.1, pp.74–83, Jan. 2002.

[7] M. P. C. Fossorier, et al., "Modeling Bit Flipping Decoding Based on Nonorthogonal Check Sums With Application to Iterative Decoding Attack of McEliece Cryptosystem," *IEEE Trans. Information Theory*, Vol.53, No.1, pp.402–411, Jan. 2007.

[8] R. B. Tremaine, et al., "IBM Memory Expansion Technology," *IBM J. Res. & Dev.*, Vol.45, No.2, pp.271–285, March 2001.