一般化低密度パリティ検査符号に対する効率的な符号化法

情報数理応用研究　　5209C028-9　寺本賢一
指導教員　後藤正幸

Efficient Encoding of Generalized Low-Density Parity-Check Codes

TERAMOTO Kenichi

1　序論

さまざまな情報システムに深く依存している現代情報社会では，情報の信頼性の確保が必要であり，そのため誤り訂正符号の構成法と復号法の重要性が増している．

誤り訂正符号の中でも低密度パリティ検査 (LDPC) 符号 [1] は，繰り返し復号を行うことで優れた復号性能を示すことが知られている．さらに R.M. Tanner は LDPC 符号に対して，高い誤り訂正能力をもつ線形ブロック符号を組み合わせる一般化 LDPC（GLDPC) 符号を提案し，従来の LDPC 符号よりも優れた性能をもつことを示した [2]．これらの符号は，受信器において確率伝播型復号法を用いれば，符号長 N に対して線形時間で復号を行うことが可能である．

一方送信器では，与えられた情報系列から必要となるパリティ系列を付加することで符号化を行うが，通常は生成行列を用いた演算により実行される．しかし，一般に生成行列は疎な行列にはならないため，多くの乗算が必要となり，符号化計算回数は $O(N^2)$ となる [1]．そのため，長い符号長の LDPC 符号や GLDPC 符号に対し符号化を実行すると，計算回数が非常に大きくなる問題が生じていた．この問題を解決するために LDPC 符号では生成行列を使わずに，既知の検査行列と情報系列より効率よく未知のパリティ系列を求める効率的な符号化法が提案された．Richardson は行・列置換を行って三角行列を構成することで求めるべき逆行列の大きさを小さくし，符号化の際に必要な計算回数を $O(N)+\alpha^2 O(N^2), 0<\alpha\ll 1$ で符号化可能な方法が提案した [1]．また，extended Irregular Repeat Accumulate (eIRA) 符号では，検査行列の一部の行列を階段状に構成することで，$O(N)$ で符号化が実行できる [2]．しかし，eIRA 符号のような行列は構成出来る符号のパラメータが限られる．これに対し，J.Lu らは符号語全体に情報ビットとパリティビットのビット位置を決定するラベル付けを行うことで，確率的に構成された任意の LDPC 符号に対して $O(N)$ で符号化が可能な方法を提案した [3]．

一方，GLDPC 符号は LDPC 符号に対して線形ブロック符号を組み合わせるため，[1] や [2] のような符号化は困難である．そこで，GLDPC 符号に対してベースとなる LDPC 符号の構造と組み合わせる線形ブロック符号の割り当て方を考慮することで，検査行列内に三角行列を構成する符号化法が提案されている [4]．しかし，依然として逆行列を求めるべき部分行列が存在し，部分的に二乗オーダーの計算回数が必要である．また，[2] では eIRA 符号と同様の手順を用いて符号化を行うために，一部の行列を確定的に配置することで線形時間で符号化が可能な手法が提案されているが，構成できる符号のクラスが限られるという問題も継承してしまう．

本研究では確率的に構成された任意の GLDPC 符号を対象とした符号化法を提案する．符号化を行うあたって，前処理として Lu らの手法と同様のラベル付けを行うために行置換・列置換を行い，GLDPC 符号でも情報ビットと既知となったパリティビットから未知のパリティビットを計算可能とする方法を示す．ここで，組み込む線形ブロック符号と未知のパリティビットの位置によっては，GLDPC 符号の検査行列の要素符号が正則行列にならず，容易にラベル付けが行えないが，擬似逆行列を用いることで全てのパリティビットを効率的に算出可能とする符号化法を提案する．提案した符号化法の最大符号化計算回数の評価式を導出し，[1] と同様に検査行列より効率的な符号化が実行可能であることを示す．

2　準備

2.1　LDPC 符号と GLDPC 符号

LDPC 符号は符号長 N，情報記号数 K に対し，非零要素が非常に少ない $M(=N-K)$ 行 N 列の，疎な検査行列 H_{LDPC} により定義される符号である．H_{LDPC} に対して，符号語 $\boldsymbol{x}=(x_1,x_2,\cdots,x_N)^{\mathrm{T}}\in F_2^N$ は，$H_{\mathrm{LDPC}}\cdot\boldsymbol{x}=\boldsymbol{0}$ と全ての行で検査方程式を満足する．ここで，F_2 は 2 元ガロア体上の要素 $\{0,1\}$ を表し，ベクトル $\boldsymbol{x}^{\mathrm{T}}$ はベクトル $\boldsymbol{x}$ の転置を表す．H_{LDPC} の各行の 1 の数 (行重み) を L，各列の 1 の数 (列重み) を J と表す．本研究では，行重みと列重みが均一な LDPC 符号を対象とし，(N,L,J) LDPC 符号と呼ぶ．また，$M=N(1-\frac{J}{L})$ で与えられる．

GLDPC 符号は，(N,L,J) LDPC 符号と符号長 n，情報系列の長さ k，最小距離が d である (n,k,d) 線形ブロック符号を組み合わせて構成される．ブロック符号のパリティ系列の長さは $m=n-k$ である．元となる LDPC 符号をベース符号と呼び，その検査行列を $H_{\mathrm{base}}=[H_{\mathrm{base},i,j}]$, $i=1,2,\ldots,M$, $j=1,2,\ldots,N$, と表す．また，ベース符号の行ベクトルを $\boldsymbol{h}_i$ とすると，$H_{\mathrm{base}}=[\boldsymbol{h}_1^{\mathrm{T}},\boldsymbol{h}_2^{\mathrm{T}},\cdots,\boldsymbol{h}_M^{\mathrm{T}}]^{\mathrm{T}}$ と表せる．H_{base} に対して、組み合わされるブロック符号を要素符号と呼び，GLDPC 符号の検査行列は $H_{\mathrm{GLDPC}}=[H_{\mathrm{GLDPC},i,j,p}]$, $i=1,2,\ldots,M$, $j=1,2,\ldots,N$, $p=1,2,\ldots,m$, で表す．したがって，要素符号の符号長である n と，ベース符号の行重み L は $L=n$ を満足することがわかる．H_{GLDPC} は H_{base} 内の要素 “1” を要素符号からランダムに選んだ長さ $(n-k)$ の列ベクトルに置き換え，要素 “0” を全零の長さ $(n-k)$ の列ベクトルに置き換えることで構成される．なお，H_{base} の各行における n 個の要素 “1” には，要素符号内の同じ列ベクトルは割り当てないものとする．このような列ベクトルによる置き換えを行拡大と呼ぶ．各行に同じ (n,k,d) 線形ブロック符号を組み合わせると H_{GLDPC} の検査記号長は $M_G=M\cdot(n-k)$ となり，符号化率 R は $R\geq R'=1-\frac{M_G}{N}$ を満足する．ここで，R' は設計符号化率である．また $0<R'<1$ より，$J<L=n$ となる．

ベース符号の i 番目の行に割り当てる要素符号の検査行列を $H_{\mathrm{comp},i}$ とする．$H_{\mathrm{comp},i}$ 内の行ベクトルを $\boldsymbol{h}'_{i,p}$ とすると，$H_{\mathrm{comp},i}=[\boldsymbol{h}'^{\mathrm{T}}_{i,1},\boldsymbol{h}'^{\mathrm{T}}_{i,2},\ldots,\boldsymbol{h}'^{\mathrm{T}}_{i,m}]^{\mathrm{T}}$ と表す．H_{base} 内の i 番目の行において，ビット位置集合を以下のように定義する．

$$\mathcal{N}(i)=\{j|\ H_{\mathrm{base},i,j}=1, j=1,2,\ldots,N\}. \qquad (1)$$

$\boldsymbol{x}$ の部分系列 $\boldsymbol{x}_i$ は $\boldsymbol{x}_i=\{x_{i,j}|j\in\mathcal{N}(i), j=1,2,\ldots,n\}$ で与えられる．すなわち，$\boldsymbol{x}_i$ は i 行目のベース符号の検査

行列における要素符号の符号語にあたる．GLDPC 符号の符号語 $\boldsymbol{x}$ は LDPC 符号と同様に $H_{\mathrm{GLDPC}} \cdot \boldsymbol{x} = \boldsymbol{0}$ を満足し，$\boldsymbol{x}_i$ は $H_{\mathrm{comp},i} \cdot \boldsymbol{x}_i = \boldsymbol{0}$ を満足する．ここで，$\boldsymbol{x}_i$ のパリティ系列に対応する部分系列を $\boldsymbol{x}_i^{\mathrm{P}} = (x_{i,1}^{\mathrm{P}}, x_{i,2}^{\mathrm{P}}, \cdots, x_{i,m}^{\mathrm{P}})^{\mathrm{T}}$，情報系列に対応する部分系列を $\boldsymbol{x}_i^{\mathrm{I}} = (x_{i,1}^{\mathrm{I}}, x_{i,2}^{\mathrm{I}}, \cdots, x_{i,k}^{\mathrm{I}})^{\mathrm{T}}$ とする．同様に，$H_{\mathrm{comp},i}$ のパリティ系列に対応する部分行列を $H_{\mathrm{comp},i}^{\mathrm{P}}$，情報系列に対応する部分行列を $H_{\mathrm{comp},i}^{\mathrm{I}}$ とする．なお，$H_{\mathrm{comp},i}$ は必ずしも組織型の検査行列ではないこととする．以上より，$\boldsymbol{x}_i = ((\boldsymbol{x}_i^{\mathrm{P}})^{\mathrm{T}}, (\boldsymbol{x}_i^{\mathrm{I}})^{\mathrm{T}})^{\mathrm{T}}$，$H_{\mathrm{comp},i} = [H_{\mathrm{comp},i}^{\mathrm{P}}, H_{\mathrm{comp},i}^{\mathrm{I}}]$ が成り立つことは自明である．このとき，$H_{\mathrm{comp},i} \cdot \boldsymbol{x}_i = \boldsymbol{0}$ であるから，パリティビット $\boldsymbol{x}_i^{\mathrm{P}}$ は，$H_{\mathrm{comp},i}^{\mathrm{P}} \cdot \boldsymbol{x}_i^{\mathrm{P}} = H_{\mathrm{comp},i}^{\mathrm{I}} \cdot \boldsymbol{x}_i^{\mathrm{I}}$，を満足する．ここで，

$$\boldsymbol{b}_i = (b_{i,1}, b_{i,2}, \cdots, b_{i,m})^{\mathrm{T}} = H_{\mathrm{comp},i}^{\mathrm{I}} \cdot \boldsymbol{x}_i^{\mathrm{I}}, \tag{2}$$

とすると，$\boldsymbol{x}_i^{\mathrm{P}}$ は $\boldsymbol{b}_i$ と $H_{\mathrm{comp},i}^{\mathrm{P}}$ の連立一次方程式 $H_{\mathrm{comp},i}^{\mathrm{P}} \cdot \boldsymbol{x}_i^{\mathrm{P}} = \boldsymbol{b}_i$ を解いて得られる．$H_{\mathrm{comp},i}$ 内の p 行目の行ベクトルにおける非ゼロのビット位置集合を以下のように定義する．

$$\mathcal{N}_{i,p} = \{j |\ H_{\mathrm{GLDPC},i,j,p} = 1, j = 1, 2, \ldots, N\}. \tag{3}$$

$\boldsymbol{b}_i$ の p 番目のビット $b_{i,p}$ は $\boldsymbol{h}_{i,p}$ と $\boldsymbol{x}_{i,p} = \{x_{i,j} | j \in \mathcal{N}_{i,p}, j = 1, 2, \ldots, N\}$ より計算できる．本稿では，行列 A 内の要素 “1” の数を $C(A)$，符号化実行の際の符号化計算回数の評価にあたって，排他的論理和の演算が行われた回数を用いる [3].

2.2 Lu らの手法 [3]

従来，LDPC 符号に対して線形時間で符号化を行うために，[2] のように検査行列内の一部の要素 “1” を確定的に配置した．J.Lu らは与えられた検査行列のタナーグラフ [1] に対して，pseudo-tree と encoding stopping set と呼ばれる 2 つの木構造で表現できるグラフに分解を行い，符号語全体に対して情報ビット，パリティビットまたは再評価ビットのラベル付けを行った．タナーグラフを階層構造で表現した pseudo-tree は再帰的に未知のパリティビットを求める label-and-decide アルゴリズムにより線形時間で符号化が可能である．更に pseudo-tree 以外の階層構造で表現される encoding stopping set に対しては，label-and-decide アルゴリズムのみでは求められないパリティビットを再評価ビットとして再度求める label-decide-recompute アルゴリズムを用いることで，線形時間で符号化が可能である．以上より，グラフの分解を行った 2 種類のグラフがいずれも線形時間で符号化が可能で，$\bar{L}$ を行重みの平均値のとき符号語全体では $4 \cdot M \cdot (\bar{L} - 1)$ で符号化可能なことを示した．

3 提案手法

3.1 提案手法の着眼点

GLDPC 符号に対する効率的な符号化法は [3], [4] などで扱われているが，いずれもベース符号 H_{base} の一部の要素 “1” の配置と，要素符号の組み込み方を確定した行列を対象としており，符号化可能なパラメータが限られていた．本研究では確率的に構成された任意の GLDPC 符号に対して効率的に符号化可能なアルゴリズムを提案する．符号化では各行で連立一次方程式 $H_{\mathrm{comp},i}^{\mathrm{P}} \cdot \boldsymbol{x}_i^{\mathrm{P}} = \boldsymbol{b}_i$ を再帰的に解くが，$H_{\mathrm{comp},i}^{\mathrm{P}}$ が正則行列のときはガウスの消去法から得られる逆行列を用い，非正則行列のときは擬似逆行列を用いて $\boldsymbol{x}_i^{\mathrm{P}}$ を求める．そのために，前処理として Lu らと同様に符号語 $\boldsymbol{x}$ に情報ビットとパリティビットの位置を決定するラベル付けを行う．具体的には，GLDPC 符号で Lu らのグラフ上における分解アルゴリズムを行列の行置換・列置換のみで表現するために，Dulmage-Mendelsohn(DM) 分解 [5] を利用して，与えられた検査行列をいくつかのブロック行列に分割を行う．また，$H_{\mathrm{comp},i}^{\mathrm{P}}$ が非正則行列となったときは方程式として制約を満たすように正則行列とは異なるラベル付けを行う．

なお，最小距離 d が検査記号数 m 以下の要素符号を組み込む場合は $H_{\mathrm{comp},i}^{\mathrm{P}}$ は正則行列と非正則行列の両方の場合が考えられる．しかし，要素符号が偶重み符号やリードソロモン符号など最大距離分離符号のときは $d = m + 1$ を満足するので，いかなるラベル付けを行っても $H_{\mathrm{comp},i}^{\mathrm{P}}$ は正則行列となる．前述の通り，LDPC 符号は要素符号に偶重み符号を組み込んだ GLDPC 符号であり，$H_{\mathrm{comp},i}^{\mathrm{P}}$ は常に正則行列であるため，逆行列を求めることができる．

3.2 提案手法のパリティビット $\boldsymbol{x}_i^{\mathrm{P}}$ の導出方法

i 番目の行で既知となったビット $\boldsymbol{x}_i^{\mathrm{I}}$ と $H_{\mathrm{comp},i}$ を式 (2) より $\boldsymbol{b}_i$ を計算し，未知の $\boldsymbol{x}_i^{\mathrm{P}}$ を求める．ここで $H_{\mathrm{comp},i}^{\mathrm{P}}$ が正則行列の場合と非正則行列の場合で計算方法が異なるため，次節以降で詳しく述べる．

3.2.1 $H_{\mathrm{comp},i}^{\mathrm{P}}$ が正則行列の場合の導出方法

連立一次方程式 $H_{\mathrm{comp},i}^{\mathrm{P}} \cdot \boldsymbol{x}_i^{\mathrm{P}} = \boldsymbol{b}_i$ より $\boldsymbol{x}_i^{\mathrm{P}}$ を求める．$H_{\mathrm{comp},i}^{\mathrm{P}}$ は正則行列のとき，逆行列 $(H_{\mathrm{comp},i}^{\mathrm{P}})^{-1}$ が存在する．式 (2) より $\boldsymbol{b}_i$ を計算し，次式より $\boldsymbol{x}_i^{\mathrm{P}}$ を求める．

$$\boldsymbol{x}_i^{\mathrm{P}} = (H_{\mathrm{comp},i}^{\mathrm{P}})^{-1} \cdot \boldsymbol{b}_i. \tag{4}$$

このときの符号化計算回数を補題 1 に示す．

補題 1 $H_{\mathrm{comp},i}^{\mathrm{P}}$ が正則行列のとき，$C(H_{\mathrm{comp},i}) - 2 \cdot m \leq$ 符号化計算回数 $\leq m \cdot (n - 2)$，を満足する． □

証明 符号化実行にあたって $H_{\mathrm{comp},i}^{\mathrm{P}}$ の逆行列 $(H_{\mathrm{comp},i}^{\mathrm{P}})^{-1}$ は予め求めておけばよいため，符号化を実行する際に必要な式 (2) と (4) の計算回数を評価すればよい．式 (2) の計算回数の上限値は $m \cdot (k - 1)$ で，式 (4) の計算回数の上限値 $m \cdot (m - 1)$ である．よって合計は $m \cdot (n - 2)$ となる．

$H_{\mathrm{comp},i}^{\mathrm{P}}$ が単位行列，または三角行列の場合，$(H_{\mathrm{comp},i}^{\mathrm{P}})^{-1}$ を用いず再帰的にパリティビットを求めることができ [2]，符号化計算回数の下限値となる． □

ここで，$m = 1$ のときは $H_{\mathrm{comp},i}^{\mathrm{P}}$ が偶重み符号であり，LDPC 符号と同じである．このとき $H_{\mathrm{comp},i}^{\mathrm{P}} = [1]$ となり逆行列が存在し，符号化計算回数は $n - 2$ で求めることができ，label-and-decide アルゴリズムと等価である [3].

3.2.2 $H_{\mathrm{comp},i}^{\mathrm{P}}$ が非正則行列の場合の導出方法

$H_{\mathrm{comp},i}^{\mathrm{P}}$ が非正則行列のとき，式 (4) ように $(H_{\mathrm{comp},i}^{\mathrm{P}})^{-1}$ を求めることが出来ない．そこで $\boldsymbol{x}_i^{\mathrm{P}}$ を求めるため，$H_{\mathrm{comp},i}^{\mathrm{P}}$ を擬似逆行列が利用できる構造 A_i に変形し，方程式を満たすように正則行列とは異なるラベル付けを行う．$\mathrm{rank}(A)$ は行列 A の階数を表し，本節では簡単のため $\mathrm{rank}(H_{\mathrm{comp},i}^{\mathrm{P}}) = m - 1$ とするが，より一般的に $\mathrm{rank}(H_{\mathrm{comp},i}^{\mathrm{P}}) < m - 1$ のときは本節で説明する操作を $m - \mathrm{rank}(H_{\mathrm{comp},i}^{\mathrm{P}})$ 回繰り返すことで $\boldsymbol{x}_i^{\mathrm{P}}$ を算出できる．

まず，$H_{\mathrm{comp},i}^{\mathrm{P}}$ は非正則行列を仮定しているため，以下の補題が成り立つ．

補題 2 m 行 m 列の非正則行列を $A_i = [\boldsymbol{a}_{i,1}^{\mathrm{T}}, \boldsymbol{a}_{i,2}^{\mathrm{T}}, \cdots, \boldsymbol{a}_{i,m}^{\mathrm{T}}]^{\mathrm{T}}$ とする．$\boldsymbol{a}_{i,r}$ を A_i 中の r 行目の長さが m の行ベクトルを表す．A_i は非正則行列なので，ガウスの消去法で，

$$\sum_{r \in \mathcal{R}} \boldsymbol{a}_{i,r} = \boldsymbol{0}, \tag{5}$$

となる行のインデックス集合 $\mathcal{R}$ が得られる．$\mathcal{R}$ のうち任意で $\tilde{r}$ 行目を取り除くと，

$$\sum_{r \in \mathcal{R} \setminus \{\tilde{r}\}} \boldsymbol{a}_{i,r} \neq \boldsymbol{0}, \tag{6}$$

となる． □

$H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列のときに $\boldsymbol{x}^{\mathrm{P}}_i$ を求めるために擬似逆行列を用いる．擬似逆行列は以下の補題で定義できる．

補題 3 m' 行 m 列 $(m' > m)$ で rank $(A'_i) = m$ の縦長の長方形行列 A'_i と $\boldsymbol{b}'_i = (b_{i,1}, b_{i,2}, \cdots, b_{i,m'})^{\mathrm{T}}$ が与えられた下で，次式の連立一次方程式を解き $\boldsymbol{x}^{\mathrm{P}}_i$ を求める．

$$A'_i \cdot \boldsymbol{x}^{\mathrm{P}}_i = \boldsymbol{b}'_i, \tag{7}$$

式 (7) に左から行列 A'^{T}_i をかけて次式を得る．

$$(A'^{\mathrm{T}}_i \cdot A'_i) \cdot \boldsymbol{x}^{\mathrm{P}}_i = A'^{\mathrm{T}}_i \cdot \boldsymbol{b}'_i, \tag{8}$$

$A'^{\mathrm{T}}_i \cdot A'_i$ は m 行 m 列の正則な正方行列となるため，式 (4) と同様に，逆行列 $(A'^{\mathrm{T}}_i \cdot A'_i)^{-1}$ を求めて，

$$\boldsymbol{x}^{\mathrm{P}}_i = (A'^{\mathrm{T}}_i \cdot A'_i)^{-1} \cdot A'^{\mathrm{T}}_i \cdot \boldsymbol{b}'_i, \tag{9}$$

より，$\boldsymbol{x}^{\mathrm{P}}_i$ を得る． □

$H^{\mathrm{P}}_{\mathrm{comp},i}$ は正方行列なので，補題 3 をそのまま適用することは出来ない．そこで，$H^{\mathrm{P}}_{\mathrm{comp},i}$ と $\boldsymbol{b}_i$ を A'_i と $\boldsymbol{b}'_i$ に変形を行う．$A_i := (H^{\mathrm{P}}_{\mathrm{comp},i})^{\mathrm{T}}$ として補題 2 の式 (6) より，線形従属となっている列ベクトルのうち任意の 1 つの $\tilde{r}$ 番目の列ベクトルを取り除くと，残りは線形独立となることがわかる．$H^{\mathrm{P}}_{\mathrm{comp},i}$ の列ベクトルを線形独立とするために，$m+1$ 行目に $\tilde{r}$ 番目が “1”でそれ以外の要素が “0”の長さが m の行ベクトルを追加して行列 A'_i を得る．同時に式 (7) の制約を満たすために，$\boldsymbol{b}'_i = (b_{i,1}, b_{i,2}, \cdots, b_{i,m}, x^{\mathrm{P}}_{i,\tilde{r}})^{\mathrm{T}}$ とする．

ここで式 (9) を用いて $\boldsymbol{x}^{\mathrm{P}}_i$ を計算するが，A'_i には線形従属となる行ベクトルが存在する．そこで補題 2 で $A_i := (H^{\mathrm{P}}_{\mathrm{comp},i})$ とし，$H^{\mathrm{P}}_{\mathrm{comp},i}$ 中の線形従属である行ベクトルを探索する．$H^{\mathrm{P}}_{\mathrm{comp},i}$ の行ベクトルと $\boldsymbol{b}_i$ は対応するので，補題 2 より線形従属となる行のインデックス集合 $\mathcal{R}$ より，$\sum_{r\in\mathcal{R}} b_{i,r} = 0$ となる．しかし $\boldsymbol{b}_i$ は $\boldsymbol{x}^{\mathrm{I}}_i$ より計算するため，$\sum_{r\in\mathcal{R}} b_{i,r} = 1$ となることがあり制約を満たさない．この場合，$b_{i,r},\ r \in \mathcal{R}$ に影響を与えるビット $x_{i,\alpha}$ を反転する．$x_{i,\alpha}$ はアルゴリズム 1 より得られる．アルゴリズム 1 では $b_{i,r}$ の計算に必要な奇数個表れた情報ビットを，H_{base} 中の i 行目以下で $\boldsymbol{x}$ の情報ビットとラベル付けされたビットから探索する．これは，偶数個では $x_{i,\alpha} + x_{i,\alpha} = 0$ となることより $\sum_{r\in\mathcal{R}} b_{i,r}$ を反転できないために，奇数個とする．

[アルゴリズム 1]

step1) $v := i$ として，初期集合 S に $b_{i,r},\ r \in \mathcal{R}$，の計算に必要なビット位置を追加する．

step2) $v := v + 1$ とする．S 中に $\boldsymbol{x}^{\mathrm{P}}_v$ に対応するビット位置が含まれる場合，$\boldsymbol{x}^{\mathrm{P}}_v$ は $\boldsymbol{b}_v$ より計算できるため，計算に必要なビット位置 S_v が求まる．

step3) S から Step 2) で表れたパリティビットを取り除いた上で，$S := S \cup S_v$ とする．

step4) S 中に $x_j,\ j = 1, 2, \ldots, N$，が偶数個あるとき，これらの x_j を全て削除する．

step5) S の中で全てが情報ビットならば，S 中のビットを任意に 1 つ選び $x_{i,\alpha}$ として終了する．パリティビットが含まれていた場合，Step 2. へいく． □

$x_{i,\alpha}$ は情報系列に依存して決定されるので，パリティビットとみなす．また，$x_{i,\alpha}$ の初期値は $x_{i,\alpha} := 0$ として，式 (2) より $\boldsymbol{b}_i$ を計算する．$\sum_{r\in\mathcal{R}} b_{i,r} = 1$ のとき，$x_{i,\alpha} := \sum_{r\in\mathcal{R}} b_{i,r}$ とすることで，$\sum_{r\in\mathcal{R}} b_{i,r} = 0$ となり，制約を満たすことができる．$H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列の最大の符号化計算回数を補題 4 に示す．

補題 4 $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のとき，最大の符号化計算回数は $2 \cdot m \cdot (n-2) \cdot i$ となる． □

証明 符号化実行にあたっては正則行列と同様に $(A'^{\mathrm{T}}_i \cdot A'_i)^{-1} \cdot A'^{\mathrm{T}}_i$ の導出と $x_{i,\alpha}$ の探索は予め求めておけばよい．よって，符号化には式 (2) と (9) の計算及び $x_{i,\alpha}$ の反転操作のみとする．ここで，A'_i は $H^{\mathrm{P}}_{\mathrm{comp},i}$ に最大で $m - (d-1)$ 個の行ベクトルを追加して得られる．よって $H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列のときの計算回数は，式 (2) の計算回数 $m \cdot (k-1)$ と，式 (9) の計算回数 $m \cdot (m + (m-d+1) - 1)$ の合計であり，計算回数の最大値は $2 \cdot m \cdot (n-2)$ となる．さらに $x_{i,\alpha}$ の反転に要した最大の計算回数は $2 \cdot m \cdot (n-2) \cdot (i-1)$ となるので，最大の符号化計算回数は $2 \cdot m \cdot (n-2) \cdot i$ となる． □

3.3 情報ビットとパリティビットのラベル付け方法

Lu らの提案したタナーグラフから木構造で表現できるグラフに分解するアルゴリズムは，大規模疎行列の連立一次方程式を行置換，列置換を行うことで解く DM 分解と同様の操作で実行できる [5]．この結果，q 個の部分行列 B_t を得る．B_t への分解方法と，各行 i において長さが k の既知の系列 $\boldsymbol{x}^{\mathrm{I}}_i$ から長さが m の未知の系列 $\boldsymbol{x}^{\mathrm{P}}_i$ を符号化可能なラベル付け方法をアルゴリズム 2 に示す．また，Lu らの手法と同様に B_t 内でラベル付けが出来ない行はアルゴリズム 3 より得られた再評価ビットを用いる．

[アルゴリズム 2] ベース符号の各行 i で $n-m$ 個の既知のビットから m 個の未知のパリティビットを導出できるよう行列の置換を行う．

step1) $i := 0$，$t := 0$ として，$l_i := N$，$H_i := H_{\mathrm{base}}$，$B_t := \phi$ とする．

step2) 行列 H_i の $M-i$ 行から l_i 列内で最小の行重み w_i をもつ行 $\boldsymbol{h}_i$ を探索し，$M-i$ 行目と交換を行う．この行列を H'_i とする．また，$B_t := B_t \cup \boldsymbol{h}_i$ とする．

step3) $w_i < m$ のとき，再評価ビットをアルゴリズム 3 より探索して，$t := t+1$ として Step7) へいく．それ以外の場合は，Step4) へいく．

step4) H'_i の $M-i$ 行目の最後の w_i 列に要素 “1”が配置されるように列置換を行う．この際に，任意に m 個選択し x^{P}_i に対応するビット位置を決定する．決定にあたっては，選択されたビット位置に対応する $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列となるようにする．残った $w_i - m$ 個のビットには情報ビットとラベル付けする．

step5) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のときは Step5-1) を，非正則行列のときは Step5-2) を実行する．

step5-1) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のとき，ガウスの消去法より $(H^{\mathrm{P}}_{\mathrm{comp},i})^{-1}$ を求める．

step5-2) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列のとき，$H^{\mathrm{P}}_{\mathrm{comp},i}$ の列ベクトルが線形独立となる行ベクトルを追加した行列 A_i を用意する．式 (9) より，$(A'^{\mathrm{T}}_i \cdot A'_i)^{-1} \cdot A'^{\mathrm{T}}_i$ を求める．アルゴリズム 1 を実行し，$x_{i,\alpha}$ を探索する．

step6) $H_{i+1} := H'_i$ とし，$l_{i+1} := l_i - w_i$，$i := i+1$ とする．

step7) 全ての行に対してラベル付けを行ったら終了する．$i \le M$ なら Step2) を再度実行する． □

アルゴリズム 2 は行列の列置換・行置換の操作のみを行っているため，本質的には置換前と等価な符号を生成している．置換を行った結果，q 個のブロック $B_t,\ t = 1, 2, \ldots, q$ を構成する．ここで，B_t では，H_{base} 中の列重みが $J > 1$ となることより，ラベル付けが出来ない行が $J-1$ 行存在する．これらの行に対して m 個のパリティビットをラベル付けするために，Lu らの手法と同様に再評価ビットを用いる．ただし，Lu らの再評価ビットは要素符号が偶重み符号のみを対象としていたが，アルゴリズム 3 では (n, k, d) 線形ブロック符号に対して再評価ビットを求めることが出来る．アルゴ

リズム 3 では，ラベル付けがされなかった $\boldsymbol{h}'_{i,p}$ に対して実行する．なお，アルゴリズム 3 はアルゴリズム 1 の Step1) と Step5) を以下のように変更することで得られる．

[アルゴリズム 3] $H_{\mathrm{comp},i}$ の i 行目の p 番目の検査方程式 $\boldsymbol{h}'_{i,p}$ を反転させる情報ビットの集合を探索する．

step1) $v := i$ として，初期集合 S に $\boldsymbol{h}'_{i,p}$ の計算に必要な $\boldsymbol{x}_i$ の集合を追加する．

step5) S の中で全てが情報ビットなら任意に一つ選んだビットを再評価ビットとして終了する．パリティビットが含まれていた場合，Step 2. を実行する． □

アルゴリズム 2 は符号構成のための前処理として扱われるので，符号化計算回数の評価には算入されない．

3.4 提案する符号化法

H_{GLDPC} に対する符号化法をアルゴリズム 4 で示す．

[アルゴリズム 4] 各行 i で情報系列と既知となったパリティ系列から，未知のパリティ系列を導出する．

step1) (前処理)H_{GLDPC} に対してアルゴリズム 2 を実行し，$\boldsymbol{x}$ に情報ビット，パリティビット，再評価ビットと $x_{i,\alpha}$ ラベル付けする．

step2) $i := M$，$t := 0$ とする．

step3) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が再評価ビットより符号語を求めるとき，再評価ビットを反転し，B_t 内のパリティビットを計算する．$t := t+1$ ととして，Step6) を実行する．それ以外は Step4) へいく．

step4) 既に判明している符号語ビット $\boldsymbol{x}^{\mathrm{I}}_i$ から式 (2) より $\boldsymbol{b}_i$ を計算する．

step5) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のとき Step 5-1) を，非正則行列のときは Step 5-2) を実行して，$\boldsymbol{x}^{\mathrm{P}}_i$ を計算する．

step5-1) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のとき，式 (4) を計算する．

step5-2) $H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列のとき，式 (7) を満たすように $x_{i,\alpha}$ を反転させて，式 (9) を計算する．

step6) $i = 0$ ならば，符号語 $\boldsymbol{x}$ を出力し，終了する．それ以外の場合，$i := i-1$ として，Step3) へ戻る． □

次節でアルゴリズム 4 を実行したときの最大の符号化計算回数を示す．

4 符号化法と符号化計算回数の評価と考察

4.1 符号化計算回数の評価方法

アルゴリズム 2 を実行して，$H^{\mathrm{P}}_{\mathrm{comp},i}$ が非正則行列となった行数を g として，H_{GLDPC} の最大の符号化計算回数を定理 1 に示す．

定理 1 与えられた H_{GLDPC} に対して，アルゴリズム 4 を実行したときの最大の符号化計算回数は $m\cdot(n-2)\cdot(M+2g^2)$ となる． □

証明 アルゴリズム 2 より，H_{GLDPC} を q 個のブロック B_t, $t = 1,2,\cdots,q$ に分けた．また，B_t は M_t 個の行ベクトルがあり，その中に g_t 行の非正則行列 $H^{\mathrm{P}}_{\mathrm{comp},i}$ が存在する．ここで，$M = \Sigma_{t=0}^{q} M_t$ であり，$g = \Sigma_{t=0}^{q} g_t$ とする．$H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列のときは補題 1，非正則行列のときは補題 4 を用いて，B_t 内の符号化計算回数の上界は次式のようになる．

$$
\begin{aligned}
&m\cdot(n-2)\cdot(M_t-g_t-(J-1))+2m\cdot(n-2)\cdot g_t^2\\
&+m\cdot(n-1)\cdot(J-1)\\
=&m\cdot(n-2)\cdot(M_t-g_t+2g_t^2)+m\cdot(J-1)\\
=&m\cdot(n-2)\cdot(M_t+2g_t^2)+m((J-1)-(n-2)g_t)\\
\le&m\cdot(n-2)\cdot(M_t+2\cdot g_t^2). \qquad (10)
\end{aligned}
$$

よって，H_{GLDPC} の符号化計算回数は $m\cdot(n-2)\cdot(M+2\cdot g^2)$ となる． □

提案した符号化法の符号化法計算回数の評価式は M と g は符号長に従い大きくなり，n と m は要素符号の符号長と検査行列の行数なので，N の値によらないと考えられる．よって符号化計算回数は $O(N)+O(g^2)$ となり，[1] のクラスの符号化法といえる．

4.2 考察

前処理では，Lu らは $\boldsymbol{x}$ のラベル付けを行う際に pseudo-tree と encoding stopping set となる集合を一度定義した後，符号化を実行した．これに対して本研究では $\boldsymbol{x}$ のラベル付けを行う際に，DM 分解を用いた．この結果，Lu らは一度の処理で複数の検査方程式を計算が出来たのに対して，提案した符号化法では 1 つ検査方程式毎に再帰的に計算する方法となった．また，[2] で用いた H_{GLDPC} に対してアルゴリズム 4 を実行した場合，$g = 0$ より線形で符号化が可能となる．符号化実行のときも $H_{\mathrm{comp},i}$ が単位行列のため逆行列を求めることなくパリティビットを再帰的に求めることができる．

$H_{\mathrm{comp},i}$ 及び $(H_{\mathrm{comp},i})^{-1}$ 内の非零要素の数は半分となることが尤もらしい [2]．よって，実際の符号化計算回数は定理 1 で示した符号化計算回数の半分程度になると期待できる．符号化において，$O(g^2)$ は H_{GLDPC} の非正則行列の行数であり，アルゴリズム 2 を実行するときに出来るだけ多くの $H^{\mathrm{P}}_{\mathrm{comp},i}$ が正則行列となるようにラベル付けをすることで符号化計算回数は小さくできる．なお，筆者が $N = 56, M = 2$ のベース符号に (14,11,3) ハミング符号を組み込んだ GLDPC 符号で確認したところ，g は M のおよそ 1 割程度となった．実際は要素符号の最小距離 d の値により，g の値は変わる．

5 まとめと今後の課題

本研究ではランダムに構成された GLDPC 符号に対して，前処理により情報系列とパリティ系列のラベル付けを行い，符号化実行の際は検査行列より再帰的に符号化を行う方法を提案した．提案した符号化法の計算回数の最大値を導出した．

今後の課題として，より少ない計算回数で符号化可能なラベル付け及び符号化方法の導出し，符号長に対して線形時間で符号化を行えることが望ましい．また本研究ではベース符号の行重みと列重みが均一な GLDPC 符号を対象としていたが，不均一な場合の符号化法も課題である．また，符号語ビットにも一般の線形ブロック符号を割り当てた Doubly-Generalized LDPC 符号に対する効率的な符号化法の検討も今後の課題として挙げられる．

参考文献

[1] T.J. Richardson and R.L. Urbanke, “Efficient encoding of low-density parity-check codes,” *IEEE Trans. Inform. Theory*, Vol. 47, No. 2, pp. 638–656, Feb. 2001.

[2] 寺本賢一, 細谷剛, 後藤正幸, 平澤茂一, “一般化 LDPC 符号に対する部分符号の構造を利用した効率的な符号化法,”電子情報通信学会技術研究報告, Vol.109, No.143, IT2009-10, pp.25–30 2009 年 7 月.

[3] J. Lu and J.M.F. Moura, “Linear time encoding of LDPC codes,” *IEEE Trans. Inform. Theory*, Vol. 56, No. 1, pp. 233–249, Jan. 2010.

[4] T. Zhang and K.K. Parhi, “A class of efficient-encoding generalized low-density parity-check codes,” *Proc. 2001 IEEE International Conference on Commun.* (ICASSP ’01), Salt Lake City, U.S.A., May 2001.

[5] T. Shibuya, “Encoding of linear codes on the rearrangement of block-triangularized parity-check matrices,” *IEICE Technical Report*, Vol. 110, No. 205, IT2010-45, pp. 69–74, Sep. 2010.