ハミング符号
Hamming Code

「符号化技術」実験

Dec 2018

1　概要

ハミング符号（Hamming Code）とはデータの誤りを検出・訂正できる線型誤り訂正符号のひとつ。

1950 年にベル研究所のリチャード・ハミングによって考案された。知られている誤り訂正符号の中では最も古く、ブロックあたり 1 ビットの誤りを訂正できる。リード・ソロモン符号などに比べると、ある程度高速で処理できるが、訂正力は高くない。このためエラー発生率が低く、速度が要求される用途に使う。ECC メモリや RAID 2 などに使用される。また、WinRAR のリカバリレコードにも使用されている [要出典]。

2　基本概念

一般にハミング符号は、ある整数 m に対し、

- 符号長 :$n = 2^m - 1$
- 情報数 :$k = n - m$

で構成される。ここで情報数とは元のデータのビット数、符号長とは生成される符号のビット数である。$m = 3$ の場合は $n = 7$、$k = 4$ となり、4 ビットのビット列を 7 ビットの符号語に置き換えるハミング符号が形成される、この場合を (7,4) ハミング符号という。ハミング符号は検査行列と生成行列と呼ばれる二つの行列を用いて処理が行われる。なお各行列計算で用いられる加算は全て排他的論理和であることに注意する。まず検査行列について述べる。この行列は m 行 n 列の行列で、全ての列要素がゼロではなく、かつ相違であ

るという条件がある。(7,4) ハミング符号の検査行列 H の一例を以下に示す。

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

検査行列 H の条件は上記に記したように全ての列が相違であり、なおかつ 0 しかない列が存在しないことである。H の列の数は n であるので、H の各列には 0 以外の全てのビットパターンが存在することになる。列の順番は任意であるが特に

$$H = [A\ I_m] \tag{2}$$

の形で表される行列の場合を組織符号という、ここで A は任意の行列、I_m は $m \times m$ の単位行列である。ハミング符号に於いては組織符号の理論上の重要性は持たないが後述する生成行列の計算が容易になるのと、符号の装置化が簡略になるので良く用いられる。なお上記の例での H も組織符号である。

次に生成行列について述べる。生成行列 G とは以下の条件を満たす非零の行列である。

$$HG^T = GH^T = 0 \tag{3}$$

ここで、G^T は G の転置行列を表す。組織符号の場合、G は以下の形で表される。

$$G = [I_k A^T] \tag{4}$$

上記の例で示した H に対する G は以下のようになる。

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{5}$$

2.1 符号化

符号化では生成行列 G と送信したいデータ $m_1, \cdots, m_k$ との乗算を行う。先ほどの例で取り上げた G を生成行列として、ビット列 1 0 1 1 を符号化する場合を考える。このとき生成される符号は

$$\begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$

となり、1 0 1 1 1 0 0 が生成された符号である。

2.2 誤り訂正と復号

復号は検査行列 H と受信したデータの積を求めることで行われる。今、受信データ Y を受け取ったとする。このとき Y に誤りが存在しない場合は

$$Y = x \cdot G \tag{7}$$

であるといえる。ここで x とは符号化される前のビット列である。この Y と H の転置行列との積を求めると H と G の関係式より

$$Y \cdot H^T = x \cdot (G \cdot H^T) = 0 \tag{8}$$

と求められる。すなわち受信語と検査行列の積が零ベクトルであるなら誤りが無いことになり、非零であるなら誤りを含むことになる。次に Y が 1 ビットの誤りを含むとする、ここで Y を以下のように仮定する。

$$Y = x \cdot G \oplus e_i \tag{9}$$

ここで e_i とは i 番目のビットが 1、それ以外のビットが 0 のビット列である。このような e_i のことを誤りベクトルという。先ほどと同様に Y と H の転置行列との積を求めると以下のようになる。

$$Y \cdot H^T = (x \cdot G \oplus e_i) \cdot H^T = x \cdot (G \cdot H^T) \oplus e_i \cdot H^T = e_i \cdot H^T \tag{10}$$

ここで e_i は i 番目のビットのみ 1 であるので、すなわち H の i 番目の列が出力されることになる。よってこの出力が H の何列目と一致するかを比べることにより誤りの位置が検出され、その部分のビットを反転することで誤りを訂正できる。例として上記の符号語が送信され、1 1 1 1 1 0 0 が受信されたとする。この受信データは 1 1 1 1 1 0 0 に誤りを含む。このデータを復号する。受信データと H の転置行列の積を求めると以下のようになる。

$$\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 0 & 0\end{bmatrix} \cdot \begin{bmatrix}1 & 0 & 1 & 1 & 1 & 0 & 0\\ 1 & 1 & 0 & 1 & 0 & 1 & 0\\ 0 & 1 & 1 & 1 & 0 & 0 & 1\end{bmatrix}^T = \begin{bmatrix}0 & 1 & 1\end{bmatrix} \tag{11}$$

この出力を転置すると、H の 2 列目と同じ値であることがわかる。よって誤りは 2 列目であり、2 列目のビットを反転させると 1 0 1 1 1 0 0 となり、元の符号語に訂正されたことがわかる。訂正後、符号語からもとの情報を取り出す。これは生成行列内に単位行列の列成分となる部分がある場合、その箇所と同位置の符号語のビットは元の情報のビットがそのまま出力されているので、これを取り出すことで元の情報が得られる。組織符号の場合は生成行列の前部分が単位行列と一致するので、単に符号語の前 k ビットを取り出すだけである。

2.3 拡張ハミング符号

(7,3) ハミング符号の場合に符号語に 2 ビットの誤りが含まれている場合を考えてみる。この時位置 i と j に誤りが含まれているとすると

$$Y = x \cdot G \oplus e_i \oplus e_j \tag{12}$$

となり、検査行列との積を取ると

$$Y \cdot H^T = e_i \cdot H^T \oplus e_j \cdot H^T \tag{13}$$

となる。この時 $e_i \neq e_j$ であるため、上記の式は検査行列の任意の 2 つの列の値排他的論理和が出力される。検査行列の定義より列の値は全て相違となるので 2 ビット誤りの時も検査行列との積が零ベクトルになることはない。ただし、別の検査行列の値と一致するため誤訂正を起こすことになる。すなわち単純なハミング符号で誤り検出のみを行うと 2 ビットの誤りまで検出できる。しかし、1 ビット誤り（訂正可能）か 2 ビット誤り（訂正不可能）であるかを判断する術がないため、この二つの機能を同時に実装することができない。そこでハミング符号に符号語全体のパリティビットを付加することで 1 ビットの誤り訂正と同時に、2 ビットまでの誤りの検出を行うことが可能になる（SECDED, single error correction, double error detection）。これを拡張ハミング符号という。