符号理論入門　－デジタルの数学－

理学部 数理科学科 原田 昌晃

はじめに

私たちの身の回りでは色々な場面でデジタル通信が用いられています. 例えば, 音楽の再生やコンピュータのデータ保存のために用いられるコンパクト・ディスクなどのデータのやり取りにはデジタル通信が用いられています. このデジタルの世界に欠かせない数理科学の一つの分野が符号理論と呼ばれるものです. ある種の情報をデジタルの通信路を通して伝える際に, 雑音が発生し受信者は間違った情報を受け取る可能性があります. このとき, 符号を用いれば少ない誤りであれば自動的に誤りを訂正して正しい情報として受信することが可能になります. この講義では, 具体的にハミング符号 e_7 を用いてどのように誤りが訂正されるのかなどを出来るだけ易しく説明したいと思います.

符号の定義

まず $\mathbb{F}_2 = \{0, 1\}$ という集合を考えます. この $\mathbb{F}_2$ に四則演算 $+, \times, (-, \div)$ を定義し, 例えば $1 + 1 = 0$ と定義します. よって, $+, \times$ は次のようになります：

$+$	0	1
0	0	1
1	1	0

$\times$	0	1
0	0	0
1	0	1

この集合 $\mathbb{F}_2$ はこの演算のもとで位数 2 の有限体とよばれます.

$\mathbb{F}_2$ の元を n 個並べたもの $(x_1, x_2, \ldots, x_n)$ はベクトルとよばれ, ここで x_i は 0 か 1 を表します. このベクトルの全体の集合は $\mathbb{F}_2^n$ と表されます. $\mathbb{F}_2^n = \{(x_1, x_2, \ldots, x_n) \mid x_i \in \mathbb{F}_2\}$ となります.

定義　C を $\mathbb{F}_2^n$ の部分集合とします, つまり, C は $\mathbb{F}_2^n$ のベクトルの幾つかの集まりを意味します. ここで, $\vec{x} = (x_1, \ldots, x_n), \vec{y} = (y_1, \ldots, y_n)$ のとき和を $\vec{x} + \vec{y} = (x_1 + y_1, \ldots, x_n + y_n)$ と定義します. このとき, C の全てのベクトル $\vec{x}, \vec{y}$ の和 $\vec{x} + \vec{y}$ が C のベクトルであるとき, C を**符号**とよびます.

定義　$\vec{x} = (x_1, x_2, \ldots, x_n), \vec{y} = (y_1, y_2, \ldots, y_n)$ に対して $\vec{x}$ と $\vec{y}$ の**距離** $d(\vec{x}, \vec{y})$ を $x_i \neq y_i$ である座標 i の個数で定義します.

定義　符号 C において 2 つの異なるベクトル $\vec{x}, \vec{y}$ の距離の中で最小となるものを C の**最小距離**とよびます.

実は, 符号 C のベクトルの個数は必ず 2^k となることが分かっています. 最小距離が d となる符号のことを $[n, k, d]$ 符号とよぶことにします.

ハミング符号 e_7

色々な符号が知られていますが, ここでは一つの有名な符号であるハミング符号 e_7 を紹介したいと思います. この符号は次のようなベクトルの集まりで定義されます：

$(0,0,0,0 \mid 0,0,0)$, $(1,0,0,0 \mid 0,1,1)$, $(0,1,0,0 \mid 1,0,1)$, $(0,0,1,0 \mid 1,1,0)$,
$(0,0,0,1 \mid 1,1,1)$, $(1,1,0,0 \mid 1,1,0)$, $(1,0,1,0 \mid 1,0,1)$, $(1,0,0,1 \mid 1,0,0)$,
$(0,1,1,0 \mid 0,1,1)$, $(0,1,0,1 \mid 0,1,0)$, $(0,0,1,1 \mid 0,0,1)$, $(1,1,1,0 \mid 0,0,0)$,
$(1,1,0,1 \mid 0,0,1)$, $(1,0,1,1 \mid 0,1,0)$, $(0,1,1,1 \mid 1,0,0)$, $(1,1,1,1 \mid 1,1,1)$.

定理 1. e_7 は $[7,4,3]$ 符号.

通信路のモデルと復号法

e_7 を用いた通信のモデルを考えることにします：

$$\boxed{\text{送信者}} \to \boxed{\text{送信ベクトル } \vec{x}} \xrightarrow[(\text{誤り } \vec{e})]{\text{通信路}} \boxed{\text{受信ベクトル } \vec{y}=\vec{x}+\vec{e}} \to \boxed{\text{受信者}}$$

1. 送信者は送信したい文字 (列) をベクトルに変換します. e_7 のベクトルの個数は１６個なので, １６個の文字への対応を例えば $A=(0000), B=(0001), \ldots, P=(1111)$ とします.

2. e_7 のベクトルの中で最初の４つの数字が一致するベクトルを選びます. 例えば (0001) の場合は e_7 のベクトル $(0001|111)$ を送信ベクトルとします.

3. 誤りの発生する可能性のある通信路を通じて, 送信ベクトルを送信します.

4. 受信者は受け取ったベクトルに誤りがある場合は訂正を行ないます. この訂正を行なう方法を復号法とよびます. 復号法は色々と知られていますが, ここでは一番簡単な次のような方法を考えたいと思います.

   - 受信ベクトル $\vec{y}$ が C のベクトルであれば送信ベクトルは $\vec{y}$ だと判断します.
   - $\vec{y}$ が C のベクトルでなければ $d(\vec{x},\vec{y})$ が最小となる C のベクトル $\vec{x}$ を探して, 送信したベクトルは $\vec{x}$ だと判断します. つまり, $\vec{y}$ に一番近い C のベクトル $\vec{x}$ に訂正された訳です.

5. 最後に受信者は誤りを訂正された受信ベクトルを文字列に変換します.

定理 2. C が $[n,k,d]$ 符号のとき C は $[(d-1)/2]$ 個の誤りを訂正可能.

したがって e_7 は1個の誤りを訂正出来ることが分かります. では, 実際に次のベクトルを受信した場合に, 正しいベクトルに訂正してみましょう.

$$\begin{aligned}
(1,0,1,0,1,1,1) &\longrightarrow (\qquad\qquad) \\
(1,0,1,1,0,0,0) &\longrightarrow (\qquad\qquad) \\
(1,0,1,0,0,0,1) &\longrightarrow (\qquad\qquad)
\end{aligned}$$

時間の都合で実用されている部分についての解説が出来ませんが, 実際にバーコード（例えば携帯でよく使われているＱＲコードを含めて）やＩＳＢＮ（国際標準図書番号）など, 身の回りに符号理論（コード理論）が使われている場面がたくさんあります.