1.6.2 符号化 ••• なんでもかんでも文字列で表わす

Ω を対象の集合とし，Σ を有限アルファベットとする．Ω から Σ^* への単射 ($x \neq y \Longrightarrow \sigma(x) \neq \sigma(y)$ を満たす写像) σ のことを**符号**と言い*，$\sigma(x)$ を $x \in \Omega$ の**符号語**(**コード**) と呼ぶ．逆に，$\alpha \in \Sigma^*$ から $\sigma^{-1}(\alpha) \in \Omega$ を求めることを**復号**という†．

〔例 1.29〕 さまざまな符号化

(1) 自然数は，基数を何にするかによっていろいろな符号化ができる．例えば，10 進数として表現する場合には $\{0,1,\ldots,9\}^*$ の元が符号語となるし，2 進数として表現する場合には $\{0,1\}^*$ の元が符号語となる．また，自然数 n を 1^n (1 進数) で表わす方法もある．

例えば，自然数 n を 2 進数表現した符号語を $\overline{n}$ で表わす (ただし，0 は λ で，0 以外は $1\{0,1\}^*$ の元で表わすものとする) と，

$$\overline{0} = \lambda,\ \overline{1} = 1,\ \overline{2} = 10,\ \ldots,\ \overline{23} = 10111,\ etc.$$

であり，奇数の集合は言語 $\{1\}\{0,1\}^*\{1\}$ で表わされる．

(2) 整数の集合は，例えば 10 進数表現するなら，$\{+,-,\lambda\}\{0,1,\ldots,9\}^+$ の部分集合として表わすことができる．

(3) コンピュータの内部ではすべての文字は 8 ビット (あるいは 16 ビット) の 2 進数で表わされている．文字と 8 ビットパターンとの対応のさせ方には何種類かの規格がある．その中の 1 つであるASCII **コード**と呼ばれる規格では

$$\mathrm{A} \leftrightarrow 01000001,\quad \mathrm{B} \leftrightarrow 01000010,\quad \mathrm{C} \leftrightarrow 01000011,\quad \ldots$$

のように定まっている．例えば 01000011 01000010 01000010 01000001 は CBBA を表わす．この符号では各文字の符号語長が一定である (等長符号).

(4) 今ではほとんど使われなくなったが，Morse(モールス)符号 (モールス信号) も 2 文字 (短点 ・と長点 –) を用いた符号化の一例である：

$$\mathrm{A} \leftrightarrow \cdot\,-,\quad \mathrm{B} \leftrightarrow -\cdots,\quad \mathrm{C} \leftrightarrow -\cdot-\cdot,\quad \ldots$$

モールス符号は文字によってコード長が異なる可変長符号である (問 1.85 参照).

---

*実際には，関数の形で示すことはしない.

†符号化 encoding．復号 decoding．符号語 (コード) code あるいは code word.
σ が単射であることは，復号が一意的にできるために課している条件である．

（5） 例 4.1 に示したような図形をグラフという．グラフも次のように符号化することができる．まず，点 (グラフ理論では「頂点」と呼ぶ) $v_1 \sim v_5$ を 2 進数 $\overline{1} \sim \overline{5}$ で表わす．点 v_i と点 v_j を結ぶ辺を文字列 $\overline{i}2\overline{j}$ で表わせば，このグラフは頂点 $v_1 \sim v_5$ と辺 v_1v_2, v_1v_4, v_2v_3, v_3v_4, v_4v_5 の集まりであると考えられるので，文字列 (G の符号語)

$$\overline{5}2\overline{1}2\overline{2}2\overline{1}2\overline{4}2\overline{2}2\overline{3}2\overline{3}2\overline{4}2\overline{4}2\overline{5} = 10121210212100210211211210021002101$$

で表わすこともできるし，$\{0,1,2\}^*$ 上の言語

$$\{\overline{5},\ \overline{1}2\overline{2},\ \overline{1}2\overline{4},\ \overline{2}2\overline{3},\ \overline{3}2\overline{4},\ \overline{4}2\overline{5}\}$$

で表わすこともできる．最初の $\overline{5}$ は点の個数を表わし，2 は文字列の中で区切り記号の役割を果たしている．

与えられたグラフ G がある条件 $\mathcal{P}$ を満たすか否かを判定する ‘問題’ も，言語

$$L := \{\overline{G} \mid G \text{ は } \mathcal{P} \text{ を満たす}\}$$

として表わすことができる．ここで，$\overline{G}$ は G の符号語 ($\in \{0,1,2\}^*$) である．G が $\mathcal{P}$ を満たすか否かは $\overline{G} \in L$ であるか否かを判定することに帰着される．

この例のように複雑な対象を符号化するには，まず基本的ないくつかの要素の符号語を決めてから，もっと複雑な要素を基本的要素の列によって表わし (同時にその符号語を決め)，それを用いてさらに複雑な要素を表わしてその符号語を決める $\cdots$ ，という方法をとるとよい．

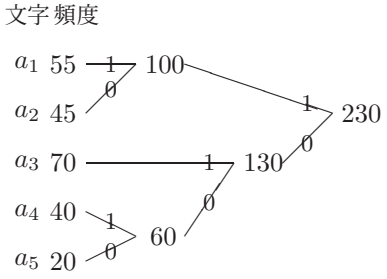
（6） 今日のようにデジタル化の時代では，アナログな対象も，近似により離散化したものを (多くの場合，0,1 の) 文字列で表現することが行なわれている．例えば，画像は平面を格子に分割して各格子点における濃淡を数値化することにより，また，音声などの波動は細分した各時刻における振幅の系列によって表わすことができ，これらはさらに 0, 1 によりデジタル化される．これも一種の符号化である． □

● コードの長さを短くするには 対象 x (Ω の元) を符号化する際，符号化したデータの総量を小さくするためには，符号語 $\sigma(x)$ の長さはできるだけ短い方がよい (データ量が小さいことはデータを送信するときや記録しておくとき

に重要)．例えば，例 1.29 (5)のようにグラフを符号化すると，点 v_i の符号語 $\sigma(v_i) = \overline{v_i}$ の長さは $\leqq \log_2 i + 1$ であるから，G の点の個数を n，辺の個数を m とすると，G の符号語長は $(\log_2 n + 2) + m(2\log_2 n + 1)$ 以下である．一方，v_i の符号語を $\sigma(v_i) := \overbrace{11\cdots1}^{i}$ と定義すると G の符号語長は最悪の場合 $(n+1) + m(2n+1)$ となり，その差は大きい．

できるだけ符号語長を短くする手法はいろいろ研究されているが，そのうちの 1 つである**Huffman(ハフマン)符号**(D.A.Huffman, 1952) は，出現頻度の高い対象ほど符号語長が短くなるようにしようという考えにもとづいている．実は，英文モールス符号も標準的な英文における各文字の出現頻度をもとに符号化されており、よく出現する文字ほど符号語長が短い．

n 個の文字があるとき，これらを出現頻度順に並べ，最も頻度の低い 2 つを除き，その代わりに仮想的な文字を 1 つ考えて，除いた 2 文字の頻度の和をその仮想文字の頻度として，この仮想文字を付け加えた $n-1$ 個の文字を次に考える．このことを，文字が 1 個になるまで繰り返す．この過程で次の図のように 0,1 を割り当てる：



各文字について，図の右端から左端まで枝に沿ってたどった道の上に現われる 0,1 を並べたものをその文字の符号語とする．上の例では，

a_1 の符号語は 11
a_2 の符号語は 01
a_3 の符号語は 10
a_4 の符号語は 001
a_5 の符号語は 000

である．この方法だと使用頻度の高いものほど短い符号語が割り当てられる．