第 4 講 言語の種類

林　恒俊

有限言語・非有限言語

- あるアルファベット上の言語について
  - **有限言語** (**finite language**)—要素の数が有限なもの
  - **非有限言語** (**infinite language**)—要素の数に限りがないもの

  など様々な言語を考えることができる。
- 要素の数が有限な言語の取扱いは比較的に容易である。たいていの場合要素を列挙して調べればよいからである。
- 要素の数に限りがない場合の取扱いには注意が必要である。以下ではこの点に関して考察する。

すべての記号列の言語

- 例えばアルファベット $\Sigma = \{a, b\}$ とすると Σ 上のすべての記号列を含む言語は $\Sigma^* = \{\Lambda, a, b, aa, ab, ba, bb, \ldots\}$ である。
- この Σ^* の要素と自然数の間に 1 対 1 対応を与えることが可能である。すなわち
  - 記号列を与えれば対応する数がユニークに定まり
  - 数を与えれば対応する記号列がユニークに定まる

  ようにすることができる。
- これには記号列を列挙順に並べ先頭からの番号を対応づければよい。

1	2	3	4	5	6	7	$\cdots$
Λ	a	b	aa	ab	ba	bb	$\cdots$

このように要素を自然数と 1 対 1 に対応させることができる無限集合を**可算**または**可付番** (**countable**) 無限集合という。可算無限集合の例には自然数、有理数、代数的数の集合などがある。

- 実数の集合は可算ではない。

言語の数

- アルファベット Σ 上の言語は Σ^* の部分集合である。したがってこのアルファベット上のすべての言語の集合は Σ^* の部分集合の集合である。
- ある集合について、そのすべての部分集合を要素とする集合を**べき** (**冪**) **集合** (**powerset**) という。集合 L のべき集合を 2^L と表示することがある。
- 集合 L を $\{a, b, c\}$ とすると

$$2^L = \{\ \emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{c, a\}, \{a, b, c\}\}$$

である。すなわち部分集合の要素は元集合の要素の有無の組合わせによって構成される。したがって有限集合 L のべき集合の要素数 $|2^L| = 2^{|L|}$ になるためこのように表示される。

- あるアルファベット Σ 上のすべての言語の集合は 2^{Σ^*} と表示される。この 2^{Σ^*} すなわちすべての記号列から構成される集合のべき集合は可算ではない。これは次のような議論によって説明されている。厳密な証明は数理言語学の教科書を参照すること。

言語の集合

- このような議論の説明には背理法を使うことが多い。
- 与えられたアルファベット Σ 上のべき集合が可算であり要素言語が順番づけられると仮定する。この仮定の下で i 番目の要素言語を L_i とすると対応する数の順に $L_1, L_2, \ldots, L_i, \ldots$ のように並べることができる。

- 列挙順にしたがって j 番目の記号列を a_j とすると Σ 上のすべての記号列も $a_1(=\Lambda), a_2, a_3, \ldots, a_j, \ldots$ のように順に並べることができる。
- それぞれの言語が記号列を含むか含まないを示す次のような表を作成できる。

	a_1	a_2	a_3	$\cdots$	a_j	$\cdots$
L_1	0	0	1	$\cdots$	1	$\cdots$
L_2	0	1	0	$\cdots$	1	$\cdots$
L_3	1	1	1	$\cdots$	0	$\cdots$
$\vdots$			$\cdots$			$\cdots$
L_i	0	1	1	$\cdots$	1	$\cdots$
$\vdots$			$\cdots$			$\cdots$

  すなわち i 番目の言語 L_i が j 番目の記号列 a_j を含む場合は 1 を、含まない場合には 0 を記入する。
- ここで表の対角線上の値に注目し次のような言語 L を構成する。すなわち L は対角線上の値が 0 ならその記号列を含み、1 ならその記号列を含まない。
- このように構成された L は $L_1, L_2, L_3, \ldots$ のすべてと異なっている。なぜなら i 番目の言語とは i 番目の記号列 a_i を含むか含まないかの点で異なるからである。
- これは前提と矛盾する。すべての言語を含むように表を作成したにもかかわらず、表にない言語が存在するからである。
- 故にすべての言語の集合は可算ではない。

考察

このような議論の進め方を**対角線論法** (**diagonalization**) という。Cantor は実数が可算でないことを説明するために対角線論法を最初に利用した。

すべての言語にコンパイラが存在するとは限らない

- 言語の集合が可算でないことから、コンパイラと対応しない言語が存在するという結論を導くことができる。
- 例えばコンパイラ (言語処理系) は C 言語で作成することができる。C 言語のプログラムは ASCII の 92 種の文字を使って書かれているので ASCII 文字アルファベット上の記号列であり、同アルファベット上の言語の要素である。当然この言語は可算である。すなわちすべてのコンパイラは数え上げることができる。
- 言語の集合は可算ではないから対応するコンパイラがない言語が存在する。

考察

これは現実の言語やコンパイラに関する話題ではない。コンパイラの集合と言語の集合を考えれば、コンパイラの集合の要素と言語の集合の要素が 1 対 1 で対応しない。すべての言語の集合の要素にコンパイラの集合の要素が対応するとは限らないということである。

言語のクラス (種類)

- 言語はアルファベット上の記号列の集合として定義されている。この記号列の複雑さにより言語の類別が可能である。一般に記号列が単純で要素の判別を容易に行うことができる言語から判別するために大容量記憶や手間のかかる手順が必要な言語まで考えることができる。
- 様々な言語の理論や応用で最も一般的な類別は
  - **正規 (正則) 言語 (regular langage, RL)**
  - **文脈自由言語 (context free language, CFL)**
  - **文脈依存言語 (context sensitive language, CSL)**

  - **帰納可算集合** (**recursively enumerable set**)

である。この分類は要素判定に必要な記憶容量と手順数により定義されている。

- 計算機科学では正規言語と文脈自由言語が日常的に応用されている。例えば
  - 正規言語は論理回路での順序回路の設計、言語処理系での字句処理
  - 正規言語と関連が深い正規表現は様々な文字列パターン検索
  - 文脈自由言語は言語処理系の構文解析

などである。

- アルファベット $\{a, b, c\}$ 上の典型的な言語の例は

言語クラス	典型例	備考
RL	$\{a^n \mid n \geq 0\}$	$\{\Lambda, a, aa, aaa, \ldots\}$
CFL	$\{a^n b^n \mid n \geq 0\}$	$\{\Lambda, ab, aabb, aaabbb, \ldots\}$
CSL	$\{a^n b^n c^n \mid n \geq 0\}$	$\{\Lambda, abc, aabbcc, aaabbbccc, \ldots\}$

である。

- あるアルファベット上のすべての正規言語 (RL) の集合を $\mathcal{L}_{\mathbf{RL}}$ と表現することがある。記号 $\mathcal{L}$ の添字に言語クラスを指定してそのクラスの言語の集合を定義する。

言語の定義手段

- 個別の言語 (すなわちあるアルファベット上の記号列の集合) を定義するために利用されている手段は以下のようなものである。
  - 言語間の演算で言語を構成する
  - 記号列のパターンを与える
  - 要素を判別する抽象機械を構築する
  - 書換え規則を使った文法を定義する

- これらの定義手段で最初の 2 方法は比較的単純な言語クラスを定義するために利用され、残りの 2 方法はより高度な言語の定義に利用されている。