第 2 講 記号列

林　恒俊

記号

- 記号列を取扱うためにはその記号列を構成する**記号** (**symbol**) が少なくとも一つは必要である。
- 様々な記号列を構成するためには複数個の記号を使うことができる。
- 通常有限の記号の集合を前提にこの記号から作られる記号列を考察する。
- このような記号列を構成する有限な記号の集合を**アルファベット** (**alphabet**) あるいは**終端記号集合** (**terminal symbol set**) という。
- 以下アルファベットを代表するために Γ, Σ などの文字を使う。
- アルファベットの要素を $\Gamma = \{0, 1\}$, $\Sigma = \{a, b, c\}$ 等のように表現する。
- あるいは n 個の要素を含むアルファベットを $\Gamma = \{a_1, a_2, \ldots, a_n\}$ のように表現する。
- これらの表示形式は集合の便宜的な表現に従っている。
- アルファベットの要素の間にあらかじめ**順序** (**order**) を仮定することがある。
    - 順序は $\Sigma \times \Sigma$ 上で定義された関係であり、大小あるいは優先順等を抽象化している。そして記号や記号列を一定順に並べるために利用できる。
    - 順序は推移律に従わなくてはならない。すなわち順序関係を $>$ で表示すると $a > b$ かつ $b > c$ なら $a > c$ である。
    - アルファベット $\Sigma = \{a, b, c\}$ で $a < b$, $b < c$ あるいは $a < b < c$ というように仮定する。

- この順序の定義は推移律を満たす限り全く任意である。例えば $a > b > c$ と定めてもかまわない。
- 関係 $a > b, b > c, c > a$ は推移律を満たさないため順序関係ではない。
- アルファベットのすべての要素間に順序が定義されるとは限らない。すべての要素間の順序が定まっている場合を**全順序** (**total order**) という。

記号列

- 常識的には**記号列** (**string**) は記号を並べたものである。例えば列 $aaabab$ はアルファベット $\{a, b\}$ 上の記号列の例である。
- しかし記号列について厳密な議論を行うためにはそれに似合う厳密な定義が必要である。
- ここでは記号列は先頭から数えた位置の数から該当する記号を返すような関数 (マッピング) と考える。
- 記号列には列の長さが存在する。いま記号列を x で代表すると $|x|$ でその長さを表示する。$|\ \ |$ は記号列の長さを返す演算子である。
- 記号列 x で、$1 \leq i \leq |x|$ の任意の整数 i について $x(i)$ は x の i 番目の記号を示す。例えば $x = aaabab$ なら $|x| = 6$ で

$$x(1) = a, x(2) = a, x(3) = a, x(4) = b, x(5) = a, x(6) = b$$

  である。
- $|x| = 0$ の記号列を**空列** (**empty string**) といい明示的に表記する場合記号 Λ を用いる。当然定義から $|\Lambda| = 0$ である。
- (**同一性**) あるアルファベット上の記号列 x, y について次の条件が満たされるなら $x = y$ である。

$$\begin{aligned} &|x| = |y| \\ &x(i) = y(i) \quad 1 \leq i \leq |x| \end{aligned}$$

考察

厳密に記号列を定義するためには記号の連結演算とその性質を定義した上で、それを基に記号列を帰納的に定義する。

また空列を認めると記号列の表示が常にユニークでなくなるが、これは整数は常に 0 が加算された表示が可能であるのと同様である。

$$1 = 1 + 0 = 1 + 0 + 0 = 1 + 0 + 0 + 0 = \cdots$$

$$a = a\Lambda = a\Lambda\Lambda = a\Lambda\Lambda\Lambda = \cdots$$

プログラムになれた人なら記号列 x の i 番目の記号を $x[i]$ と表示した方が分かりやすいかもしれない。これは配列の要素指定と同じ表現だからである。

記号列の操作

- いくつかの記号列に関する操作 (演算) を定義する。
  - 記号列の精査
  - 記号列の連結
  - 部分列
  - その他

記号列の精査

- 任意の記号列 x について特定の位置 i に特定の記号 c があるかどうかを調べることが必要なことがある。このため関数 $x_c(i)$ を次のように定義する。

$$x_c(i) = \begin{cases} 1 & x(i) = c \text{ の場合} \\ 0 & \text{そうでない場合} \end{cases}$$

例えば記号列 $x = aaabab$ とすると $x_a(5) = 1$ である。

- 記号列 x 中の記号 c の個数を $\#_c(x)$ とすると

$$\#_c(x) = \sum_{i=1}^{|x|} x_c(i)$$

例えば $\#_a(aaabab) = 4$ である。

連結操作

- 元々記号列は記号を連結して構成される。さらに記号列を**連結** (**concatenation**) して新しい記号列を生成することができる。
- あるアルファベット上の記号列 x, y について x と y を連結した記号列 z は

$$z = x \circ y$$

と表示される。$\circ$ で連結演算子を示している。

- この $z = x \circ y$ について次の性質が成立する。これが連結演算子 $\circ$ の定義と考えられる。

$$\begin{array}{ll} |z| = |x| + |y| & \\ z(i) = x(i) & 1 \leq i \leq |x| \\ z(|x| + j) = y(j) & 1 \leq j \leq |y| \end{array}$$

- 例えば $aaab \circ ab = aaabab$
- x を任意の記号列とすると $\Lambda \circ x = x \circ \Lambda = x$ である。(証明せよ)
- $\circ$ は結合則が成立する。すなわち x, y, z を任意の記号列とすると

$$(x \circ y) \circ z = x \circ (y \circ z) = x \circ y \circ z$$

- ある記号列 x を n 回連結した記号列を x^n と表示する。

$$x^n = \underbrace{x \circ x \circ \cdots \circ x}_{n}$$

- x^n は次のように**帰納的** (**inductive**) に定義される。

$$\begin{cases} x^0 = \Lambda \\ x^{i+1} = x \circ x^i \quad i = 0, 1, 2, \ldots \end{cases}$$

- 一般に乗算演算子が代数式で省略されるのと同様に曖昧にならない場合には記号列の式の $\circ$ 演算子も省略されることがある。

考察

帰納的あるいは再帰的な定義や証明は記号に関する理論において非常に重要な位置を占めている。あらゆるところで帰納や再帰が現れるのでよく慣れるようにしよう。

部分列

- ある記号列をいくつかの部分に分解することが可能である。分解した記号列で元の記号列の
  - 先頭部を含む部分を**接頭辞**あるいは**前置部** (**prefix**)
  - 最後尾を含む部分を**接尾辞**あるいは**後置部** (**suffix**)
  - 任意の連続した部分を**部分列** (**substring**)

  という。
- 記号列 u について空列を含む任意の記号列 w があって $u = v \circ w$ である場合、v は u の接頭辞である。
- 記号列 u について空列を含む任意の記号列 v があって $u = v \circ w$ である場合、w は u の接尾辞である。
- 記号列 u について空列を含む任意の記号列 v, x があって $u = v \circ w \circ x$ である場合、w は u の部分列である。
- 例えば記号列 abc で Λ, a, ab, abc は接頭辞、Λ, c, bc, abc は接尾辞、Λ, a, b, c, ab, bc, abc は部分列である。なお記号列 ac は abc の接頭辞、接尾辞、部分列のいずれでもない。

逆列

- ある記号列 x の前後を逆にした記号列を x の**逆列** (**reversal**) といい x^R と表示する。
- $z = x^R$ とすると $z(i) = x(|x| + 1 - i) \quad 1 \leq i \leq |x|$
- $\Lambda^R = \Lambda$
- x を任意の記号列とすると $(x^R)^R = x$
- x, y を同じアルファベット上の記号列とすると $(xy)^R = y^R x^R$
- 例えば $(java)^R = avaj$
- xx^R のような形式の記号列を**回文** (**palindrome**) という。x が回文なら $x^R = x$ である。(証明せよ)

記号列の順序

- 記号に関してアルファベットの要素に順序を定義することにより順序づけることができる。
- 記号列に対しても適切な方法による順序づけが必要な場合がある。代表的な方法は
  - **辞書順** (**lexicographical order**)
  - **列挙順** (**enumeration order**)

  である。

辞書順

- 辞書順は辞書の単語の並べ方に基づく順序付け法である。
- 順序が定義されているアルファベット Σ 上の記号列を x, y とする。x と y は先頭から j の位置まで一致するものとする。すなわち

$$0 \leq j \leq \min(|x|, |y|)$$

という j について

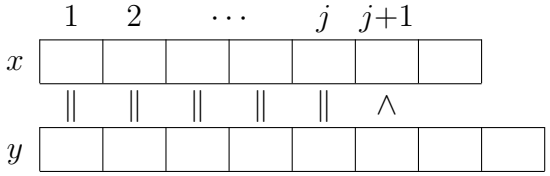
$$x(i) = y(i) \quad 1 \leq i \leq j$$

とする。このとき

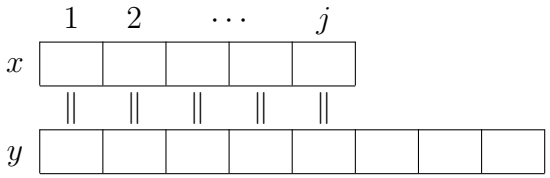
$$\begin{cases} j < \min(|x|, |y|) \wedge x(j+1) < y(j+1) & \textbf{(1)} \quad \text{または} \\ j = |x| \wedge |x| < |y| & \textbf{(2)} \end{cases}$$

なら $x < y$ である。すなわち

(1) x と y が先頭からある位置まで部分的に一致するときにはその次の位置の記号により順序が定まる



(2) x が y より短くて y の前置部と x が一致するなら $x < y$



- 例えば $\Sigma = \{a, b\}$ で $a < b$ とすると

$$\Lambda < a < aa < aaa < aaabbb < aabaaa < ab < b < ba < bb$$

列挙順

- 記号列を与えられたときそれがすべての記号列中の何番目にあたるか判断できるような順序を定義しておくと便利である。列挙順はこのような定義の一つである。
- 順序が定義されているアルファベット Σ 上の記号列を x, y とする。
  - $|x| < |y|$ なら $x < y$

  - $|x| = |y|$ なら x と y の順序は辞書順による
- すなわち記号列を長さの順に並べ、同じ長さなら辞書順に並べる。
- 例えば $\Sigma = \{a, b\}$ で $a < b$ なら

$$\Lambda < a < b < aa < ab < ba < bb < aaa < aab < \cdots$$

- 次の課題を考察するとよい。あるアルファベットについてそのアルファベット上のすべての記号列を列挙順に並べた時
  - n 番目の記号列を求める
  - 与えられた記号列が何番目になるか計算する

  方法について考察せよ。