第 13 講 正規言語のポンピング定理

林　恒俊

ポンピング定理

- FSA を調べると正規言語が満たさなければならないある条件を見つけることができる。そしてこの条件を使って正規言語でない言語を判定することができる。以下にこの条件について説明する。なおこの正規言語が満たす条件を正規言語の**ポンピング定理** (**pumping theorem**) と呼ぶことがある。
- ある正規言語 L が有限でないと仮定すると次のような条件を満たす記号列 x, y, z が存在する。
  1. $y \neq \Lambda$ かつ
  2. $i \geq 0$ を満たすすべての i について $xy^iz \in L$
- いいかえると言語の要素の記号列が十分に長ければ
  - その記号列を 3 分割して
  - その中心の部分を取去っても
  - あるいはその部分をいくら繰返しても

  その言語の要素になるようにできる。
- 例えば言語 `0(01)*1` の要素 001011 で $x = 0$, $y = 0101$, $z = 1$ と選べばよい。01, 001011, 0010101011, $\ldots$, $0(0101)^i1$ $(i \geq 0)$ はすべてこの言語の要素である。

ポンピング定理の証明

- この定理は次のように証明することができる。有限でない言語 L を受理する DFA を $M = (Q, \Sigma, \delta, q_0, F)$ とする。記号列 $w \in L$ が M

で受理される過程を考える。

$$(q_0, [1, w]) \vdash_M (q_1, [2, w]) \vdash_M \cdots \vdash_M (f, [|w| + 1, w])$$

- この過程中に出現した状態の列 $q_0, q_1, \ldots, f$ 中の状態数は $|w| + 1$ 個になる。ところが M の状態数は高々$|Q|$ 個しかない。ここで w が十分に長くて $|w| + 1 > |Q|$ と仮定すると状態列 $q_0, q_1, \ldots, f$ の中には必ず同じ状態が重複して出現する。重複する状態を q_k とすると計算過程の状態列は

  $$q_0, q_1, \ldots, q_k, \ldots, q_k, \ldots, f$$

  となる。

- この $q_k, \ldots, q_k$ の間の計算過程を取去り、その $q_k, \ldots, q_k$ 過程で受理された記号列を入力記号列から取除いた計算過程も受理する計算過程である。

- あるいは $q_k, \ldots, q_k$ の間の計算過程を任意に繰返し、その $q_k, \ldots, q_k$ 過程で受理された記号列を入力記号列に追加した計算過程も受理する計算過程である。

- したがって

  - q_0 から q_k に遷移するまでの w の先頭からの記号列を x
  - q_k から再び q_k に遷移する記号列を y
  - 最後の q_k から f まで遷移する記号列を z

  とするとポンピング定理を満たしている。

- なお多数の要素をより少ない場所に配分すると必ず重複が生じるということを**巣箱の原理** (**pigeonhole principle**) といい様々な証明で利用される。

正規言語帰属判定

- この定理は言語が正規言語でないことを証明するためによく利用される。例えば $L_{ab} = \{a^n b^n \mid n \geq 0\}$ という言語が正規言語でないことは次のように証明される。

- L_{ab} が正規言語と仮定する。記号列 $w \in L_{ab}$ に対して $w = xyz$ を適当に割当てて y の部分を任意に繰返しても L_{ab} に含まれるようにできる。

$$\underbrace{aaa\cdots a}_{x}\underbrace{aaabbb}_{y}\underbrace{b\cdots bbb}_{z}$$

  しかし

  - y を ab の両方を含むように割当てると y の部分を取去ってもよいが繰返したものは L_{ab} の要素ではない。
  - y を a または b の連続した列中に割当てると y を取去ったり繰返したものは a と b の数が異なるため L_{ab} の要素ではない。

- これは仮定と矛盾する。したがって L_{ab} は正規言語ではない。

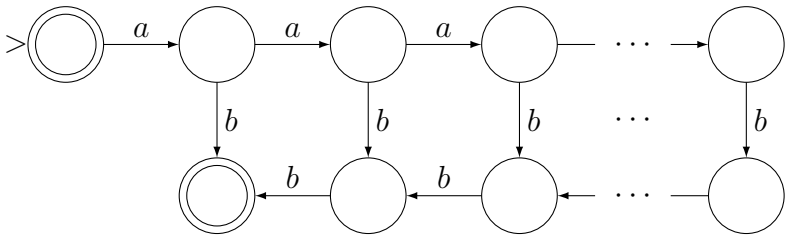
正規言語帰属判定 (その 2)

- 正規言語は和、連結、繰返し、結び、補集合などの演算に関して閉じている。すなわち正規言語にこれらの演算を行った結果も正規言語である。これを利用していくつかの言語が正規言語でないことを検証できる。
- 次の言語 $L_{a=b} = \{x \mid x \in \{a,b\}^*, \#_a(x) = \#_b(x)\}$ は正規言語でない。なおこの $L_{a=b}$ は a と b が同数含まれる記号列からなる言語である。
- この検証には背理法を利用する。すなわち $L_{a=b}$ は正規言語であると仮定する。
- $L' = \{a\}^*\{b\}^*$ は a のみの列 b のみの列がこの順に並んだ記号列からなる言語で明らかに正規言語である。
- これらの前提から $L_{a=b} \cap L'$ も正規言語である。
- $L_{a=b} \cap L'$ は a と b の数が同じでかつ ab の順に並んでいる記号列からなる言語であるから $\{a^n b^n \mid n \geq 0\}$ と同一であり正規言語ではない。(前項に矛盾)
- したがって $L_{a=b}$ は正規言語でない。

- この判定法は適用範囲が広くて様々な言語の判定に利用することができる。

有限な言語と有限でない言語

- 前述のように $a^n b^n$ 形の記号列の言語 L_{ab} は FSA で受理できない。しかし言語が有限なら受理する DFA を次のように構成することが可能である。



  記号列が有限である限りその最大長にあわせてはしご状の部分を延長すればよい。

  言換えると梯子に追加された状態が入力された a の数を記憶しているものと見なされる。

- 言語 L_{ab} が正規言語でない、すなわち FSA で受理されない理由は以下の通りである。
  - 有限でない言語 L_{ab} はいくらでも長い記号列を含んでいる。
  - それを受理する FSA は記号の長さに対応するため梯子の段を必要な状態の数だけ延長しなければならない。記号列の長さに限りがなければ梯子の段数 (つまり状態数すなわち記憶容量) をそれに見合うだけ大きくしなければならない。
  - ところが状態数に限界がない抽象機械は FSA ではない。
  - したがって FSA で L_{ab} は受理できない。
- 正規言語を構成する演算中、和演算と連結演算をいくら行っても元の集合が有限なら結果は必ず有限集合になる。

  また基礎である空集合や単 1 記号要素集合は当然有限集合である。

したがって有限でない言語を構成するためにはどこかで Kleene の $*$ 演算を使わなければならない。そして Kleene の $*$ 演算があればそこが 0 回以上の繰返しを許す部分記号列を導入する。