目次

1 反復補題で遊ぼう

坂口 和彦

1.1 はじめに

正規表現は現代の多くのプログラマにとって馴染み深い道具である。形式言語理論における正規表現[*1]は文字列の集合を表すためのものであり、正規表現で記述できる集合は正規言語 (*regular language*)[*2] と呼ばれる。また、ある文字列の集合が正規言語であることを指して単に正規であると言うことがある。

さて、n 個の 0 の並びの後に n 個の 1 の並びが続く列の集合 $\{0^n1^n : n \in \mathbb{N}\}$ は正規だろうか。実はこれは正規ではなく、他にも正規ではないような例、即ち正規表現によって表せないような集合は無数に存在する。反復補題は、このような言語が正規ではないことを証明するための道具の 1 つである。本章では、Coq 上で正規言語を扱うためのライブラリを紹介し、実際にそのライブラリの反復補題を使っていくつかの言語が正規ではないことを証明する。

1.2 正規表現と有限オートマトン

本節では、本題に入る前の準備として、いくつかの定義を導入する。

定義 **1.1** (語と言語) 文字の有限集合 Σ について、Σ の元の 0 個以上の並びを Σ の語 (*word*) と呼ぶ。Σ の語の集合を Σ の言語 (*language*) と呼ぶ。語 w の長さは $|w|$ で表記し、語 w 中の文字 a の出現数は $|w|_a$ で表記する。長さ 0 の語は ϵ で表記する。語 w の n 個の並びから成る語 $\underbrace{w \ldots w}_{n}$ を w^n で表記する。

[*1] 後方参照などを含まないため、一部の読者にとっては見慣れないものかもしれない。定義は後述する。

[*2] 本章での「言語」は文字列の集合の意味である。

1.3　反復補題

反復補題は、ある言語クラスに関してある言語がそのクラスに含まれていないことを示すための補題である。特に、これ以降で示す正規言語の反復補題 (*pumping lemma for regular languages*) は、ある言語が正規言語ではないことを示すための補題である。本章では、正規言語以外に対する反復補題は扱わないので、正規言語の反復補題を指して単に反復補題と呼ぶ。

ある言語が正規であるとすると、その言語に対応する DFA が存在する。DFA では、入力列を読んでいる途中でその DFA の状態の個数以上の情報を記憶しておくことはできない。よって、列を読んでいる途中に記憶しないといけない情報の数の上限を定められないような言語は、正規言語ではない。これが反復補題の基本的なアイディアである。

DFA M について、M 上のある状態から始めて十分に長い列 x で状態遷移させることを考える。DFA の状態は有限個なので、その状態遷移の過程である状態 q を 2 回以上通るはずである。q を通った 2 つの箇所の間にある列を繰り返したとしても、その部分の末尾では必ず状態 q に戻り、最終的に到達する状態も変化しない。これをより精密に書くことで、以下の反復補題が得られる。

定理 1.11 (正規言語の反復補題) 任意の正規言語 $A \subseteq \Sigma^*$ について、次を満たすような反復長 $p \in \mathbb{N}$ が存在する [*5]: $xyz \in A$ と $|y| \geq p$ を満たす任意の $x, y, z \in \Sigma^*$ について $u, v, w \in \Sigma^*$ が存在し、以下が全て成り立つ。

- $y = uvw$
- $v \neq \epsilon$
- 任意の $i \in \mathbb{N}$ について、$xuv^iwz \in A$

この命題を論理式で書くと、以下のようになる。

$$\forall A \subseteq \Sigma^*.\,(A \text{ is regular} \Rightarrow \exists p \in \mathbb{N}.\,\forall x, y, z \in \Sigma^*.\,(xyz \in A \wedge |y| \geq p \Rightarrow \exists u, v, w \in \Sigma^*.\,(y = uvw \wedge v \neq \epsilon \wedge \forall i \in \mathbb{N}.\,xuv^iwz \in A)))$$

[*5] 反復長は A のみに依存して決まる。

2 ドレスデンでCoqを書いて暮らす

平井 洋一

ドイツで、形式手法の仕事をしている。ドレスデンというところで働いている。ドイツの東端だ。ドレスデンにいた王様が昔はポーランドの一部を治めていた。いまは Volks Wagen の工場がある。ドレスデンにはたくさんの路面電車が走っている。人口が五十万人なのに、十五も路線がある。路面電車は一編成が五車両だ。人間があまりいない。百人以上の群衆を見たことがない。とくに、スーツを着ている人が、たいへん珍しい。一ヶ月暮らして二人しか見ない。市街に暴走族がいない。速度無制限の高速道路が無料だからかもしれない。コンビニが存在しない。あるデンマーク人が、日本に滞在して、夜中にコンビニに行けることを発見して、日本はハッカーにとっていいところなのだと述べていた。もちろん、つくば市並木二丁目に住んでしまうと不便なほうだったが、十五分も自転車をこげば夜中でも用が足りた。ドレスデンに住んでしまうと、どれだけ行っても夜中に店は開いていない。つくばと似ているのは、つくばの公務員宿舎みたいなものを東ドイツが国民のために建てていたので、同じ規格の団地がたくさん並んでいることだ。

ドレスデンでの仕事で、Coq を書いている。世の中には、Haskell プログラマを集めて PHP を書かせるというような冗談 (か実話) があるらしいのだが、そういうことにはなっていない。Emacs の ProofGeneral で Coq を書いている。数ヶ月に一度は成果を見せないといけないという制約のもとで、本当なら数年かかることをしているので、数ヶ月ごとに何をどう見せるかちょっと発明が必要なのだが、結局のところは Coq を書いている。

上司は Hendrik Tews という人で、オランダの [*1]Bart Jacobs のところにいたらしい。なぜか、検証対象の業界にも証明器にも詳しい人だ。ProofGeneral の plugin を書いたらしい。職場ではむやみに圏論の話をして人々を怖がらせ

[*1] ベルギーにも Bart Jacobs がいる。