決定不能問題から始める計算可能性理論入門 *

y.†

2017 年 12 月 10 日

最終更新日:2017 年 12 月 31 日

概要

決定問題とは与えられた入力に対し Yes または No で答える計算課題をいう．決定問題のうち，それを解くアルゴリズムが存在しないようなものを決定不能問題という．本稿ではアルゴリズムの数学的な定義のひとつである Turing 機械と呼ばれる計算モデルを導入し，決定問題が決定不能であることを証明するための技法を紹介する．また数論や代数学における決定不能問題や，決定不能問題に関する未解決問題についても触れる．

0 記号と定義

- 有限集合 Σ に対し，Σ^* で Σ の元を文字とする文字列[*1]全体の集合を表す．例えば，$\Sigma = \{\mathtt{a}, \mathtt{b}\}$ のとき $\Sigma^* = \{\varepsilon, \mathtt{a}, \mathtt{b}, \mathtt{aa}, \mathtt{ab}, \mathtt{ba}, \mathtt{bb}, \mathtt{aaa}, \dots\}$ となる (ここで ε は長さ 0 の文字列 (空文字列) を表す)．Σ を**アルファベット** (alphabet) と呼ぶ．
- A, B を集合とする．A の部分集合から B への関数 f のことを A から B への**部分関数** (partial function) という．本稿ではこのことを $f\colon A \rightharpoonup B$ で表す．[*2]
  $x \in A$ に対して $f(x)$ が定義されていること (すなわち x が f の定義域に含まれていること) を $f(x)\downarrow$ で表し，$f(x)$ が定義されていないことを $f(x)\uparrow$ で表す．さらに $f(x)\downarrow$ かつ $f(x) = y$ であることを $f(x)\downarrow = y$ で表す．
  f の定義域が A 全体に一致するとき f は**全域的** (total) であるという．

1 決定問題

決定問題とは，与えられた入力に対して Yes または No で答える計算課題のことをいう．厳密な定式化の前に，まずは例を見よう．

問題 1.1 (素数判定問題，primality testing).
Input: 正整数 n
Question: n は素数か？

---

* この文書は，都数の総会で発表した内容を公開用に修正したものです．

† http://iso.2022.jp/

[*1] すなわち，Σ の元の有限列全体の集合のことである．あるいは，Σ を自由基底とする自由モノイドということもできる．

[*2] 部分関数の記法には特にこれといって定着したものはないようで，文献によってまちまちである．

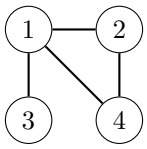
この問題は簡単に “解ける”:

解答 1.2. $n = 1$ なら No，$n = 2$ なら Yes，$n > 2$ のときは $2, 3, \ldots, n-1$ で順番に割ってみて，ひとつでも割り切れるものがあれば No，そうでないときは Yes と答えればよい．

決定問題には他にも様々なものが考えられる．ここでは 2 つだけ取り上げることにする．[*3]

- **部分和問題** (subset sum problem):
  Input: 自然数の有限集合 $S \subseteq \mathbb{N}$ と自然数 $x \in \mathbb{N}$ の組
  Question: $\sum_{n \in T} n = x$ なる部分集合 $T \subseteq S$ が存在するか？
- **Hamilton 閉路問題** (Hamilton circuit problem):
  Input: 有限無向グラフ G
  Question: G は Hamilton 閉路 (全ての頂点を一度だけ通るような閉路) を持つか？

上で述べた問題では入力の範囲が問題によって異なっており，自然数の有限集合であったり有限グラフであったりする．本稿ではこれらを全て統一的に扱うために，以降は決定問題の入力は全て文字列であると仮定する．計算の対象となるものは基本的に全て文字列で表すことができるので，この仮定は何らの悪影響ももたらさない．例えば，有限集合 $\{1, 2, 3\}$ はそのまま`{1,2,3}`と表せるし，有限グラフ



は`{{1,2,3,4},{{1,2},{1,3},{1,4},{2,4}}}`と表せる (どちらもアルファベットは $\Sigma = \{0, 1, \ldots, 9, ,, \{, \}\}$ でよい).[*4]

これを数学的な言葉で書き直せば次のようになる．

定義 1.3 (決定問題). **決定問題** (decision problem) とは，あるアルファベット Σ に対する Σ^* の部分集合[*5]，あるいはその特性関数 $\Sigma^* \to \{\text{Yes}, \text{No}\}$ のことをいう．[*6]

さきほど素数判定問題 1.1 が簡単に “解ける” と言った．ここで決定問題が “解ける” とはどういうことかを定義しておこう．

定義 1.4 (決定問題の決定可能性). $f \colon \Sigma^* \to \{\text{Yes}, \text{No}\}$ を決定問題とする．f を計算する**アルゴリズム** (algorithm) が存在するとき，f は**決定可能** (decidable) であるといい，そうでないとき**決定不能** (undecidable) であるという．

アルゴリズムは**プログラム** (program) と言い換えることもできる．f を計算するアルゴリズムが存在するとき f は**計算可能** (computable) であるという．現段階では「アルゴリズム」「プログラム」「計算可能」はま

[*3] ここで上げた 2 つの問題は実は **NP 完全問題** (NP-complete problem) と呼ばれる種類の問題であることが知られている．
[*4] この考えをさらに推し進めて，計算可能性理論では入力はひとつの自然数であるとすることも多い．
[*5] 決定問題を文字列の集合とみなすときは**言語** (language) と呼ぶこともある．
[*6] 入力が有限グラフの場合など，グラフを表していない文字列 (例えば `,1},}2{` のような) もあるのだから特性関数の定義域を文字列全体にするのはまずいのではないかと思うかもしれない．しかし「文字列がグラフを表しているかどうか」などは機械的に判定可能な条件なので問題が生じることはない．例えば，Hamilton 閉路問題は「w があるグラフ G を表しており，かつ G が Hamilton 閉路を持つ」ような文字列 w 全体の集合であるとしてよい．

だ定義されておらず，これを数学的にどう定義するかが問題になる．これについては次節で議論することにする．

決定問題を考える動機付けとして，歴史的に有名かつ興味深い決定問題を 2 つ紹介して本節を終えることにする．

問題 1.5 (Hilbert の第 10 問題, Hilbert's tenth problem (1900)).
Input: 整数係数多項式 $f(x_1, \dots, x_n) \in \bigcup_{k>0} \mathbb{Z}[x_1, \dots, x_k]$
Question: Diophantus 方程式 $f = 0$ は整数解 $(x_1, \dots, x_n) \in \mathbb{Z}^n$ を持つか？

数論において Diophantus 方程式の解の個数を数えること，特に解の個数が 0 個かどうかを判定することは非常に重要な問題である．Hilbert の第 10 問題は数論における究極的な目標のひとつ (だった) と言える．

問題 1.6 (Entscheidungsproblem, Hilbert and Ackermann (1928)).
Input: 一階述語論理の閉論理式 φ
Question: φ は任意の構造で真か？

Gödel による一階述語論理の完全性定理より，実際には上の質問は「φ は証明可能か？」と同値である．
上記の問題は 2 つとも後に否定的に解決された．

定理 1.7 (Turing (1936)). Entscheidungsproblem は決定不能である．

定理 1.8 (Matiyasevich-Robinson-Davis-Putnam の定理, Matiyasevich (1970)). Hilbert の第 10 問題は決定不能である．

Turing はまさしく定理 1.7 の証明のために Turing 機械[*7]を導入したのである．本稿では上記の 2 つの定理を証明することはしないが，次節以降，代わりにその基礎となる理論的な道具立てを与える．

2 Turing 機械と Church-Turing の提唱

前節では「アルゴリズム」「プログラム」「計算可能」などの用語を定義せずに用いた．本節ではこれらの用語を Turing 機械によって定義する．

Turing 機械は 1 本の無限に長い**テープ** (tape) と左右に動くことのできる**ヘッド** (head) からなる (図 1)．テープは正方形のマス目 (セル) に区切られており，1 つのマスには 1 つの文字が入る．ヘッドはテープの 1 つマスを見ており，左右に 1 マスずつ動くことができる．Turing 機械は内部状態を 1 つだけ保持することができ，機械は一定の規則 (遷移関数) に基づいてテープ上の文字を書き換えたりヘッドを左右に動かしたりして計算を進める．

[*7] Turing 本人は automatic machine (a-machine) と呼んでいた．

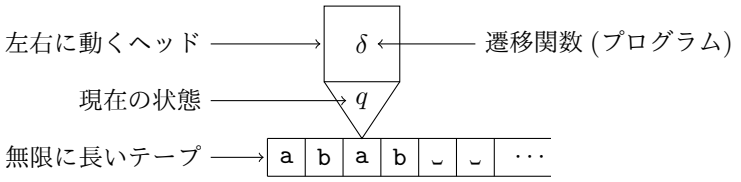


図 1　Turing 機械

Turing 機械は形式的には以下のように定義される．

定義 2.1 (Turing 機械). **Turing 機械** (Turing machine) M とは以下のデータからなる 7 つ組 $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ である:

1. **状態** (state) の集合 Q は有限集合,
2. **入力アルファベット** (input alphabet) Σ は**空白記号** (blank symbol) ␣ を含まない有限集合,
3. **テープアルファベット** (tape alphabet) Γ は空白記号 ␣ と Σ を含む有限集合,
4. **遷移関数** (transition function) $\delta\colon Q \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$, [*8]
5. **開始状態** (initial state) $q_0 \in Q$,
6. **受理状態** (accepting state) $q_{\text{accept}} \in Q$,
7. **拒否状態** (rejecting state) $q_{\text{reject}} \in Q, q_{\text{reject}} \neq q_{\text{accept}}$.

Turing 機械 M は入力 $w \in \Sigma^*$ に対して次のように計算を行う:

1. $w = w_1 w_2 \cdots w_n$ $(w_i \in \Sigma)$ とする．テープの左端から n マス目までは $w_1, w_2, \ldots, w_n$ を書き込み，それ以外の全てのマスには空白記号␣を書き込む．ヘッドはテープの左端のマスを見ている状態にし，機械の内部状態を開始状態 $q_0 \in Q$ に設定する．
2. 機械の現在の状態が $q \in Q$ でヘッドが見ている文字が $a \in \Gamma$ で，さらに $\delta(q, a) = (r, b, D)$ であるとする．このとき機械の現在の状態を q から r に変更し，ヘッドが見ているマスの文字を a から b に書き換え，$D = \mathrm{L}$ ならヘッドを左に，$D = \mathrm{R}$ なら右に 1 マスだけ移動させる．[*9]機械の現在の状態が $q_{\text{accept}}, q_{\text{reject}}$ でない限りこれを繰り返す．
3. 機械の現在の状態が q_{accept} に到達したら直ちに計算を停止して $M(w)\downarrow = $ 受理 $= \textsc{Yes} = 1$ と定める．
4. 機械の現在の状態が q_{reject} に到達したら直ちに計算を停止して $M(w)\downarrow = $ 拒否 $= \textsc{No} = 0$ と定める．
5. 機械が永遠に $q_{\text{accept}}, q_{\text{reject}}$ のどちらにも到達しない場合は $M(w)\uparrow$ と定める．

これにより M は部分関数 $M\colon \Sigma^* \rightharpoonup 2 = \{0, 1\}$ を定める．

注意 2.2 (Turing 機械の頑健性). Turing 機械の定義は次の意味で**頑健** (robust) である: 定義に以下のような変更を加えても Turing 機械の計算能力は一切変化しない．

- ヘッドの動きとして，左右に動くだけでなくその場に留まるという動作も許す．
- テープを両方向とも無限にする．
- テープを 2 本にする．

[*8] 正確には $\delta\colon (Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}) \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$ とするべきだが，表記の煩雑さを避けるためこのようにした．
[*9] ヘッドがテープの左端を見ていてかつ $D = \mathrm{L}$ のときはその場に留まるものとしておく．

- etc.

Turing 機械の計算の様子を簡単な例で確かめてみよう．

例 2.3 (Turing 機械の例). $Q := \{q_0, q_1, q_2, q_3, q_{\text{accept}}, q_{\text{reject}}\}, \Sigma := \{\mathtt{a}, \mathtt{b}\}, \Gamma := \{\mathtt{a}, \mathtt{b}, \mathtt{␣}\}$ とし，遷移関数 $\delta\colon Q \times \Gamma \to Q \times \Gamma \times \{\mathrm{L}, \mathrm{R}\}$ を以下の表のように定義する．

	a			b			␣			
q_0	q_1	␣	R	q_{reject}	b	R	q_{accept}	␣	R	(左端の a を消す)
q_1	q_1	a	R	q_1	b	R	q_2	␣	L	(右端まで行く)
q_2	q_{reject}	a	R	q_3	␣	L	q_{reject}	␣	R	(右端の b を消す)
q_3	q_3	a	L	q_3	b	L	q_0	␣	R	(左端まで行く)

これにより定まる Turing 機械 M は決定問題 $\{\, \mathtt{a}^n\mathtt{b}^n \mid n \geq 0 \,\} \subseteq \Sigma^*$ を判定する機械であり，[*10]どんな入力に対しても必ず停止する (ここで $\mathtt{a}^n := \underbrace{\mathtt{a}\cdots\mathtt{a}}_{n}$ である)．

入力 aabb に対するこの機械の動作を見てみよう．紙面の都合上 Turing 機械の図を描くわけにはいかないから，計算の各時点での Turing 機械の状態を**計算状況** (configuration) と呼ばれる文字列で表すことにする．状態の右隣に書いてある文字がヘッドが現在見ている文字である．

$$
\begin{array}{ll}
q_0\mathtt{aabb} & q_3\mathtt{␣ab␣␣} \\
\mathtt{␣}q_1\mathtt{abb} & \mathtt{␣}q_0\mathtt{ab␣␣} \\
\mathtt{␣a}q_1\mathtt{bb} & \mathtt{␣␣}q_1\mathtt{b␣␣} \\
\mathtt{␣ab}q_1\mathtt{b} & \mathtt{␣␣b}q_1\mathtt{␣␣} \\
\mathtt{␣abb}q_1\mathtt{␣} & \mathtt{␣␣}q_2\mathtt{b␣␣} \\
\mathtt{␣ab}q_2\mathtt{b␣} & \mathtt{␣}q_3\mathtt{␣␣␣␣} \\
\mathtt{␣a}q_3\mathtt{b␣␣} & \mathtt{␣␣}q_0\mathtt{␣␣␣} \\
\mathtt{␣}q_3\mathtt{ab␣␣} & \mathtt{␣␣␣}q_{\text{accept}}\mathtt{␣␣}
\end{array}
$$

これより $M(\mathtt{aabb})\downarrow = 1$ である．

Turing は人間の紙とペンを使った計算を詳細に分析することにより Turing 機械の定義を得た．したがって可能な全ての計算は Turing 機械で模倣できると考えられる (これに関する詳細な議論は Turing の原論文 [13, section 9] を参照のこと)．この信念は **Church-Turing の提唱** (Church-Turing thesis) と呼ばれ，広く信じられている．

提唱 2.4 (Church-Turing の提唱). 部分関数 $f\colon \Sigma^* \rightharpoonup \{\textsc{Yes}, \textsc{No}\}$ が計算可能 $\iff$ f を計算する Turing 機械が存在する．

この提唱の右辺は数学的に意味のはっきりした命題であるのに対し，左辺はそうではないことに注意する．これは定理というよりはむしろ左辺を右辺で定義しているのだと思った方がわかりやすい．Church-Turing の提唱の妥当性を支持する根拠として，注意 2.2 で述べた定義の頑健性や，Turing 機械が他の様々な計算モデル (ラムダ計算，再帰関数，レジスター機械など) と計算能力が等価であるという事実などが挙げられる．

Church-Turing の提唱の $\Longrightarrow$ を信じるならば，アルゴリズムを記述する際に Turing 機械の遷移関数を詳細に記述する必要はないことになる．本稿でもこの立場をとることにし，以降は遷移関数を直接記述することはしない．

[*10] ちなみに，この問題は Turing 機械より計算能力が低い**有限オートマトン** (finite automaton) では判定できないことが知られている．さらに，有限オートマトンは読み取り専用 Turing 機械と計算能力が等価であることも知られている．

注意 2.5 (決定不能問題の存在). Church-Turing の提唱から決定不能問題の存在が直ちにわかる．実際，Turing 機械の遷移関数は可算通りしかないのに対し，決定問題は Σ^* の冪集合の濃度 (これは連続体濃度である) だけあるからである．しかしながら，計算手順を文章ではっきりと示すことができるような関数が可算個しかないことは直感的に明らかだし，構成的な証明でもないので，これはとくに意味のある結果ではない．

Turing 機械 M は定義から有限のデータで構成されており，したがって文字列で表すことができる．この文字列を M の記述といい $\langle M \rangle$ で表す．$\langle M \rangle$ は例 2.3 で書いた表 (を一列に並べた文字列) を表していると思えばよい．人間は遷移関数と入力文字列が与えられれば Turing 機械の動作をシミュレートできるので，Church-Turing の提唱よりそのような Turing 機械が存在することがわかる．

定理 2.6 (万能 Turing 機械). ある Turing 機械 U が存在して，どんな Turing 機械 M と入力文字列 w に対しても $U(\langle M \rangle, w) = M(w)$ となる (部分関数として等しい).[*11] このような U を**万能 Turing 機械** (universal Turing machine) と呼ぶ．

万能 Turing 機械は現代の Neumann 型コンピューター (プログラム内蔵方式) の原型である．(ENIAC などの黎明期のコンピューターと異なり) 現代のコンピューターは目的ごとに配線を繋ぎ直したりする必要はなく，用途ごとにプログラム $\langle M \rangle$ を入れ換えればひとつのハードウェアで様々な動作をさせることができる．あるいは，U は入力されたプログラム $\langle M \rangle$ を実行するインタープリターであると思うこともできる．

注意 2.7. 上では $\Sigma^* \rightharpoonup 2$ という形の部分関数しか考えなかったが，計算理論では $\mathbb{N}^k \rightharpoonup \mathbb{N}$ という形の部分関数を考えることも多い．このような部分関数を計算させるためには，Turing 機械の定義に例えば次のような変更を加えればよい．

- $q_{\mathrm{accept}}, q_{\mathrm{reject}}$ の代わりにただひとつの停止状態 q_{halt} を用意する．
- $\Sigma = \{\mathtt{1}\}$ とする．
- 入力が $(n_1, \ldots, n_k) \in \mathbb{N}$ であるときは，テープの左端から連続する n_i 個の $\mathtt{1}$ を空白記号 $\mathtt{␣}$ で区切って並べ，残りを空白記号 $\mathtt{␣}$ で埋めた状態 $\mathtt{1}^{n_1}\mathtt{␣}\mathtt{1}^{n_2}\mathtt{␣}\cdots\mathtt{␣}\mathtt{1}^{n_k}\mathtt{␣␣}\cdots$ から計算を始める．
- q_{halt} に到達した時点でテープ上にある空白記号以外の文字の個数を出力値とする．
- q_{halt} に到達しなければ未定義とする．

もちろん $\Sigma = \{\mathtt{1}\}$ の代わりに $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ を用いて 2 進列を入力してもよいし，q_{halt} に到達した時点でテープ上に $\mathtt{1}, \mathtt{␣}$ 以外の文字が残っていたら計算結果を未定義にする，などの条件を加えてもよい．これらの細かい変更は Turing 機械の計算能力に影響を与えない．

演習問題 2.8 (Turing 機械の構成). 以下の動作をする Turing 機械の遷移関数を具体的に与えよ．ただし $\mathbb{N}^k \rightharpoonup \mathbb{N}$ の形の部分関数に関しては注意 2.7 の定義を用いること．Turing 機械のシミュレーターを用いると実際の動作を確認することができて便利である (例えば `https://turingmachinesimulator.com/` など).

1. $\Sigma = \{\mathtt{a}\}$ とし，$\{\, \mathtt{a}^n \mid n \text{ は奇数} \,\}$ を判定する機械．Hint: $|Q \setminus \{q_{\mathrm{accept}}, q_{\mathrm{reject}}\}| = 2$ で十分である．
2. $f\colon \mathbb{N} \times \mathbb{N} \rightharpoonup \mathbb{N}; f(x, y) = x + y$ を計算する機械．Hint: 自明．
3. $f\colon \mathbb{N} \rightharpoonup \mathbb{N}; f(x) = 2x$ を計算する機械．

[*11] 実際には M ごとに入力アルファベットが異なるのでこの主張は正確ではない．正確を期すためには，例えば U の入力アルファベットを $\Sigma := \{\mathtt{0}, \mathtt{1}\}$ とし，任意のアルファベットの任意の文字列を Σ^* の文字列に変換する規則を固定して考えればよい．

4. $f\colon \mathbb{N}\times\mathbb{N} \rightharpoonup \mathbb{N}; f(x,y)=\max\{x-y,0\}$ を計算する機械.
5. $\Sigma=\{(,)\}$ とし，入力 $w\in\Sigma^*$ が“対応のとれた括弧の列”かどうかを判定する機械 (たとえば `(())()` は対応のとれた括弧の列であるが `)(()` はそうではない)．Hint: 左から右に向かって走査し，閉じ括弧を見つけたら対応する開き括弧と一緒に何らかの文字 `x` で消す．これを繰り返す．
   この問題を言語とみなしたものは **Dyck 言語**と呼ばれている．

3 停止問題

本節では具体的な決定不能問題の例として停止問題を取り上げる．停止問題は決定不能性の根源であり，全ての決定問題のうちで最も重要なものである．

問題 3.1 (停止問題, Halting problem; HALT).
Input: Turing 機械の記述 $\langle M\rangle$ と入力文字列 w の組
Question: $M(w)\downarrow=1$ か？

停止問題の決定不能性は対角線論法により証明される．

定理 3.2. HALT は決定不能である．

証明. 仮に HALT が決定可能であるとすると，HALT を判定する Turing 機械 H が存在する．H が全域的であることに注意すると次のような Turing 機械 D を作ることができる:

$$D(\langle M\rangle) := \begin{cases} 1=\text{受理} & \text{if } H(\langle M\rangle,\langle M\rangle)=0,\\ 0=\text{拒否} & \text{if } H(\langle M\rangle,\langle M\rangle)=1. \end{cases}$$

このとき D も全域的であり

$$\begin{aligned} D(\langle D\rangle)\downarrow=0 &\iff H(\langle D\rangle,\langle D\rangle)=1 && (D\text{ の定義より})\\ &\iff D(\langle D\rangle)\downarrow=1 && (H\text{ の定義より}) \end{aligned}$$

となり矛盾を生じる． □

注意 3.3. 上の証明が実際に対角線論法になっていることを見よう．Turing 機械は可算個しかないので一列に並べることができ，$H(\langle M_i\rangle,\langle M_j\rangle)$ の計算結果を以下の表 1 のように一覧にすることができる．

表 1　$H(\langle M_i\rangle,\langle M_j\rangle)$ の計算結果 ($M_i(\langle M_j\rangle)\downarrow=1$ かどうか) の一覧

	$\langle M_1\rangle$	$\langle M_2\rangle$	$\langle M_3\rangle$	$\langle M_4\rangle$	$\cdots$	$\langle D\rangle$	$\cdots$
M_1	$\boxed{1}$	0	1	0	$\cdots$	1	$\cdots$
M_2	1	$\boxed{1}$	1	1	$\cdots$	1	$\cdots$
M_3	0	0	$\boxed{0}$	0	$\cdots$	0	$\cdots$
M_4	1	1	0	$\boxed{0}$	$\cdots$	1	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	
D	0	0	1	1	$\cdots$	$\boxed{?}$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\ddots$

このとき D は対角線上の値と逆の計算結果を返すように設計されているため，どの Turing 機械とも異なる部分関数を計算する．ところが D 自身は Turing 機械なのでこの表のどこかに現れなければならず，矛盾を生じる．

停止問題 HALT 自体は純粋に計算可能性理論的な決定問題であるが，HALT の決定不能性を利用することで様々な問題の決定不能性を導くことができる．これが本稿の残りの部分で行うことである．

4 Post の対応問題

本節では，Post の対応問題と呼ばれる組み合わせ論的な決定問題を題材に，具体的な問題が決定不能であることを示すための典型的な考え方を述べる．

定義 4.1 (ドミノ，マッチ). 文字列の組 (u, v) を縦に並べたもの $\left[\frac{u}{v}\right]$ を**ドミノ** (domino) と呼ぶ．ドミノの有限列

$$\left[\frac{u_1}{v_1}\right]\left[\frac{u_2}{v_2}\right]\cdots\left[\frac{u_n}{v_n}\right]$$

が**マッチ** (match) であるとは，各段の文字列をつなげてできる文字列がそれぞれ等しい ($u_1u_2\cdots u_n = v_1v_2\cdots v_n$) ことをいう．

問題 4.2 (ポストの対応問題, Post correspondence problem; PCP).
Input: ドミノの有限集合 P [*12]
Question: P はマッチを持つか？ (ただし，同じドミノを何回使ってもよいとする)

例 4.3.

$$P = \left\{ \left[\frac{\texttt{ab}}{\texttt{abab}}\right], \left[\frac{\texttt{b}}{\texttt{a}}\right], \left[\frac{\texttt{aba}}{\texttt{b}}\right], \left[\frac{\texttt{aa}}{\texttt{a}}\right] \right\}$$

は次のマッチを持つ:

$$\left[\frac{\texttt{ab}}{\texttt{abab}}\right]\left[\frac{\texttt{ab}}{\texttt{abab}}\right]\left[\frac{\texttt{aba}}{\texttt{b}}\right]\left[\frac{\texttt{b}}{\texttt{a}}\right]\left[\frac{\texttt{b}}{\texttt{a}}\right]\left[\frac{\texttt{aa}}{\texttt{a}}\right]\left[\frac{\texttt{aa}}{\texttt{a}}\right].$$

一方,

$$P = \left\{ \left[\frac{\texttt{abc}}{\texttt{ab}}\right], \left[\frac{\texttt{ca}}{\texttt{a}}\right], \left[\frac{\texttt{acc}}{\texttt{ba}}\right] \right\}$$

は (上段の方が下段より常に長いので) マッチを持たない.

PCP が決定不能であることを示すために，**Turing 還元** (Turing reduction, Turing 帰着ともいう) と呼ばれる手法を用いる: もし仮に PCP を判定するアルゴリズムが存在したら，それをもとにして HALT を判定するアルゴリズムが作れてしまうことを示す．一般に決定問題 B が決定可能であると仮定して A が決定可能であることが示せるとき，A は B に **Turing 還元可能** (A is Turing reducible to B) であるといい，$A \leq_{\mathrm{T}} B$ と書く.[*13]直感的には $A \leq_{\mathrm{T}} B$ は A より B の方が “難しい” あるいは “多くの情報を持っている” ということを表している．明らかに，A が決定不能で $A \leq_{\mathrm{T}} B$ なら B も決定不能である．

[*12] ここではアルファベットは固定せず，入力される P 内の文字列として任意の文字種を許すものとする．これを固定された入力アルファベットで実現するためには，文字の代わりに自然数の 2 進法表記を用いるなどすればよい.

[*13] Turing 還元可能性をより正確に定義するためには**神託機械** (oracle machine) の概念が必要であるが，本稿の範囲ではここで述べた定義で十分であるのでこれ以上の深入りはしない.

$\leq_{\mathrm{T}}$ は反射的かつ推移的な二項関係，すなわち**前順序** (preorder) をなす．したがって「$A \leq_{\mathrm{T}} B$ かつ $B \leq_{\mathrm{T}} A$」は決定問題の間の同値関係になり，これによる同値類を **Turing 次数** (Turing degree) と呼ぶ．次数の理論は計算可能性理論の中心的な話題の一つであるが，本稿の範囲を超えてしまうためここでは扱わない．詳細は Soare [5] などの教科書を見られたい．

定理 4.4 (Post (1946) [14]). $\mathrm{HALT} \leq_{\mathrm{T}} \mathrm{PCP}$.

証明は中間問題を導入して 2 段階に分けて行う．

問題 4.5 (修正されたポストの対応問題, Modified Post correspondence problem; MPCP**).**
Input: ドミノの有限集合 P と $d \in P$
Question: P は d を左端とするマッチを持つか？

補題 4.6. $\mathrm{MPCP} \leq_{\mathrm{T}} \mathrm{PCP}$.

証明. PCP が決定可能であると仮定する．このとき MPCP が決定可能となることを示す．MPCP の入力を

$$P = \left\{ d = \left[\frac{u_1}{v_1}\right], \left[\frac{u_2}{v_2}\right], \ldots, \left[\frac{u_n}{v_n}\right] \right\}$$

とし，$*, \Diamond$ を P に現れない文字とする．一般に文字列 $w = w_1 \cdots w_l$ に対して

$$\begin{aligned} *w &:= *w_1 * w_2 * \cdots * w_l \\ w* &:= \ w_1 * w_2 * \cdots * w_l* \\ *w* &:= *w_1 * w_2 * \cdots * w_l* \end{aligned}$$

と定義し，

$$P' := \left\{ \left[\frac{*u_1}{*v_1*}\right], \left[\frac{*u_1}{v_1*}\right], \left[\frac{*u_2}{v_2*}\right], \ldots, \left[\frac{*u_n}{v_n*}\right], \left[\frac{*\Diamond}{\Diamond}\right] \right\}$$

と定める．このとき明らかに

$$P \text{ が左端が } d \text{ であるマッチを持つ} \iff P' \text{ がマッチを持つ}$$

であるから，P' がマッチを持つかどうかを PCP を判定するアルゴリズムを用いて確かめればよい． □

補題 4.7. $\mathrm{HALT} \leq_{\mathrm{T}} \mathrm{MPCP}$.

証明. MPCP が決定可能であると仮定して，HALT が決定可能となることを示す．HALT の入力を $(\langle M \rangle, w)$ とする (ただし $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\mathrm{accept}}, q_{\mathrm{reject}}), w = w_1 w_2 \cdots w_n$ とする)．このとき，ドミノの有限集合 P を

$$M(w)\downarrow = 1 \iff P \text{ は } d \text{ を左端とするマッチを持つ} \qquad (*)$$

を満たすように構成する．そのために，P のマッチが M の受理計算履歴 (例 2.3 にあるような計算状況の列) となるようにすればよい．すなわち，“ドミノの連結で Turing 機械の動作を模倣する．”

を Γ に含まれない文字とする．

1. まず，計算の開始状況を表すドミノ $d = \left[\frac{\#}{\#q_0 w_1 \cdots w_n \#}\right]$ を P に追加する．
2. 各 $a, b \in \Gamma, q, r \in Q\,(r \neq q_{\mathrm{reject}})$ に対し $\delta(q, a) = (r, b, \mathrm{R})$ のとき $\left[\frac{qa}{br}\right]$ を P に追加する．

3. 各 $a,b,c \in \Gamma, q,r \in Q\,(r \neq q_{\mathrm{reject}})$ に対し $\delta(q,a) = (r,b,\mathrm{L})$ のとき $\left[\dfrac{cqa}{rcb}\right]$ を P に追加する．
4. 各 $a \in \Gamma$ に対して $\left[\dfrac{a}{a}\right]$ を P に追加する．
5. $\left[\dfrac{\#}{\#}\right], \left[\dfrac{\#}{\text{␣}\#}\right]$ を P に追加する．
6. 各 $a \in \Gamma$ に対して $\left[\dfrac{aq_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right], \left[\dfrac{q_{\mathrm{accept}}a}{q_{\mathrm{accept}}}\right]$ を P に追加する．
7. $\left[\dfrac{q_{\mathrm{accept}}\#\#}{\#}\right]$ を P に追加する．

この P に対し条件 $(*)$ が成り立つことを言えばよい．

$(\Longrightarrow)$ 受理状況に至る計算の履歴が存在するので，それがそのままマッチになっている．

$(\Longleftarrow)$ できあがった文字列 $\#C_0\#C_1\#\cdots\#C_r\#$ を考えると，各 $s \geq 0$ に対して「M が s ステップ目で停止していなければそのときの計算状況が C_s であること」が s に関する帰納法で証明できる．#の個数を考えるとマッチの右端は $\left[\frac{q_{\mathrm{accept}}\#\#}{\#}\right]$ でなければならないから，M は入力 w に対して必ず q_{accept} に到達する． □

例 4.8. 例 2.3 における Turing 機械 M をとる．上の証明における $(\langle M\rangle, \mathtt{ab})$ に対する P は

$$P = \left\{ d = \left[\frac{\#}{\#q_0\mathtt{ab}\#}\right], \left[\frac{q_0\mathtt{a}}{\text{␣}q_1}\right], \left[\frac{q_0\text{␣}}{\text{␣}q_{\mathrm{accept}}}\right], \left[\frac{q_1\mathtt{a}}{\mathtt{a}q_1}\right], \left[\frac{q_1\mathtt{b}}{\mathtt{b}q_1}\right], \left[\frac{q_3\text{␣}}{\text{␣}q_0}\right], \left[\frac{\mathtt{a}q_1\text{␣}}{q_2\mathtt{a}\text{␣}}\right], \left[\frac{\mathtt{b}q_1\text{␣}}{q_2\mathtt{b}\text{␣}}\right], \left[\frac{\text{␣}q_1\text{␣}}{q_2\text{␣␣}}\right], \right.$$
$$\left[\frac{\mathtt{a}q_2\mathtt{b}}{q_3\mathtt{a}\text{␣}}\right], \left[\frac{\mathtt{b}q_2\mathtt{b}}{q_3\mathtt{b}\text{␣}}\right], \left[\frac{\text{␣}q_2\mathtt{b}}{q_3\text{␣␣}}\right], \left[\frac{\mathtt{a}q_3\mathtt{a}}{q_3\mathtt{aa}}\right], \left[\frac{\mathtt{b}q_3\mathtt{a}}{q_3\mathtt{ba}}\right], \left[\frac{\text{␣}q_3\mathtt{a}}{q_3\text{␣}\mathtt{a}}\right], \left[\frac{\mathtt{a}q_3\mathtt{b}}{q_3\mathtt{ab}}\right], \left[\frac{\mathtt{b}q_3\mathtt{b}}{q_3\mathtt{bb}}\right], \left[\frac{\text{␣}q_3\mathtt{b}}{q_3\text{␣}\mathtt{b}}\right], \left[\frac{\mathtt{a}}{\mathtt{a}}\right], \left[\frac{\mathtt{b}}{\mathtt{b}}\right], \left[\frac{\text{␣}}{\text{␣}}\right],$$
$$\left.\left[\frac{\#}{\#}\right], \left[\frac{\#}{\text{␣}\#}\right], \left[\frac{\mathtt{a}q_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right], \left[\frac{\mathtt{b}q_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right], \left[\frac{\text{␣}q_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right], \left[\frac{q_{\mathrm{accept}}\mathtt{a}}{q_{\mathrm{accept}}}\right], \left[\frac{q_{\mathrm{accept}}\mathtt{b}}{q_{\mathrm{accept}}}\right], \left[\frac{q_{\mathrm{accept}}\text{␣}}{q_{\mathrm{accept}}}\right], \left[\frac{q_{\mathrm{accept}}\#\#}{\#}\right] \right\}$$

となる．読者は左端が d であるようなマッチを作ろうとすると勝手に計算が進んでしまうことを確認してみてほしい．

注意 4.9. 問題 4.2 ではドミノに用いられるアルファベットは固定しなかったが，実際にはドミノに現れる文字は $\Sigma = \{\mathtt{0},\mathtt{1}\}$ のみであると仮定してよい．実際，入力に現れうる文字の全体を $a_1, a_2, \ldots$ とすれば，ドミノに現れる a_i を全て $\mathtt{1}\underbrace{\mathtt{0}\cdots\mathtt{0}}_{i}\mathtt{1}$ に置き換えれば等価な問題になる．

5 いろいろな決定不能問題

5.1 行列に関する決定問題

問題 5.1 (Matrix mortality problem; $\mathrm{MORT}_{\mathbb{Z}}(n)$). 正整数 n を固定する．

Input: 整数成分の正方行列の有限集合 $F \subseteq \mathrm{M}_n(\mathbb{Z})$

Question: F の生成する乗法半群 $\langle F\rangle$ は零行列を含むか？

定理 5.2 (Peterson (1970) [15]). $\mathrm{MORT}_{\mathbb{Z}}(n)$ は $n \geq 3$ のとき決定不能．

証明. $\mathrm{PCP} \leq_{\mathrm{T}} \mathrm{MORT}_{\mathbb{Z}}(3)$ を示す．PCP の入力を

$$P = \left\{ \begin{bmatrix} u_1 \\ v_1 \end{bmatrix}, \dots, \begin{bmatrix} u_n \\ v_n \end{bmatrix} \right\}$$

とする．ここで，注意 4.9 と同様にして P に現れる文字は $\{2,3\}$ のみであるとしてよい．任意の文字列 $u = u_0 \cdots u_k, v = v_0 \cdots v_l \in \{2,3\}^*$ に対して

$$W(u,v) := \begin{pmatrix} 10^{|u|} & 0 & 0 \\ 0 & 10^{|v|} & 0 \\ \sigma(u) & \sigma(v) & 1 \end{pmatrix}$$

と定める．ここで $|u| = k, |v| = l$ はそれぞれ u, v の長さであり，$\sigma(u) := \sum_{i=0}^{k} u_i \cdot 10^{k-i}$ は文字列を自然数の 10 進法表記とみなしたときの値である．このように定めると例えば

$$\begin{aligned} W(23,2)W(223,32) &= \begin{pmatrix} 100 & 0 & 0 \\ 0 & 10 & 0 \\ 23 & 2 & 1 \end{pmatrix} \begin{pmatrix} 1000 & 0 & 0 \\ 0 & 100 & 0 \\ 223 & 32 & 1 \end{pmatrix} = \begin{pmatrix} 100000 & 0 & 0 \\ 0 & 1000 & 0 \\ 23223 & 232 & 1 \end{pmatrix} \\ &= W(23223, 232). \end{aligned}$$

となり，ドミノの結合を行列の積で再現することができる．さらに

$$S := \begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, T := \begin{pmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

とおき，

$$F := \{S, T, W(u_1, v_1), \dots, W(u_n, v_n), W(u_1, 1v_1), \dots, W(u_n, 1v_n)\}$$

とおく．このとき

$$P \text{ がマッチを持つ} \iff \langle F \rangle \ni 0$$

となることを示す．

($\Longrightarrow$) $\begin{bmatrix} u_{i_1} \\ v_{i_1} \end{bmatrix} \cdots \begin{bmatrix} u_{i_m} \\ v_{i_m} \end{bmatrix}$ を P のマッチのひとつとすると，$SW(u_{i_1}, 1v_{i_1})W(u_{i_2}, v_{i_2}) \cdots W(u_{i_m}, v_{i_m})T = 0$ となる．

($\Longleftarrow$) ある $W(u,v) \in \langle F \setminus \{S,T\} \rangle$ が存在して $SW(u,v)T = 0$ となり，かつ $u \in \{2,3\}^*, 1u = v$ となることを示せばよい．詳細は原論文 (たったの 3 ページ！) を参照してほしい． □

実は，$n = 2$ の場合は未解決である．

未解決問題 5.3. $\mathrm{MORT}_{\mathbb{Z}}(2)$ は決定可能か？ また，$\mathbb{Z}$ を $\mathbb{Q}$ に変えた $\mathrm{MORT}_{\mathbb{Q}}(2)$ についてはどうか？

部分的な結果として次が知られている．

事実 5.4. 以下が成り立つ．

- 入力を上三角行列 (下三角行列) のみからなる集合に限れば決定可能である ($(1,1)$ 成分が 0 の行列と $(2,2)$ 成分が 0 の行列を含むことが必要十分である).
- 単射な半群の準同型 $\Sigma^* \times \Sigma^* \to \mathrm{M}_2(\mathbb{C})$ は存在しない (Cassaigne and Harju and Karhumaki (1999) [16]). したがって $\mathrm{MORT}_{\mathbb{Z}}(2), \mathrm{MORT}_{\mathbb{Q}}(2)$ の決定不能性を示すのに PCP を利用することはできそうにない.

- 入力する行列の個数を $|F| = 2$ に限った問題 $\textsc{Mort}_{\mathbb{Q}}(2,2)$ は決定可能であるが，成分を実数にした $\textsc{Mort}_{\mathbb{R}}(2,2)$ は BSS モデル (実数を直接扱うことのできる計算モデルのひとつ) で決定不能である (Bournez and Branicky (2002) [17])．したがって $\textsc{Mort}_{\mathbb{Q}}(2,2)$ の決定可能性には $\mathbb{Q}$ の数論的な性質が本質的に用いられている．
- $\textsc{Mort}_{\mathbb{Z}}(2)$ は NP 困難である (Bell and Hirvensalo and Potapov (2012) [18])．

Matrix mortality problem において零行列を単位行列に変えた問題は Matrix identity problem と呼ばれている．

問題 5.5 (Matrix identity problem; $\textsc{Ident}_{\mathbb{Z}}(n)$[*14]). 正整数 n を固定する．
Input: 整数成分の正方行列の有限集合 $F \subseteq \mathrm{M}_n(\mathbb{Z})$
Question: F の生成する乗法半群 $\langle F \rangle$ は単位行列を含むか？

事実 5.6 (Bell and Potapov (2009) [19]). $\textsc{Ident}_{\mathbb{Z}}(n)$ は $n \geq 4$ のとき決定不能．

この証明は PCP の変種である ICP と呼ばれる決定不能問題を Turing 還元するために，$\mathrm{PSL}_2(\mathbb{Z})$ が位数 2 と 3 の群の自由積になっているという性質を利用して，自由群の直積からの単射な群の準同型 $\mathrm{FG}(\Sigma) \times \mathrm{FG}(\Sigma) \to \mathrm{SL}_4(\mathbb{Z})$ を作ることでなされた．

事実 5.7 (Potapov and Semukhin (2016) [20]). $\textsc{Ident}_{\mathbb{Z}}(n)$ は $n \leq 2$ のとき決定可能．

この証明も $\mathrm{PSL}_2(\mathbb{Z})$ が自由積になっていることを利用しており，$\mathbb{Q}$ の場合には一般化できない．
Matrix mortality problem の場合と同様に，Matrix identity problem においても $n = 3$ の場合は未解決である．

未解決問題 5.8. $\textsc{Ident}_{\mathbb{Z}}(3)$ は決定可能か？ また，$\mathbb{Z}$ を $\mathbb{Q}$ に変えた $\textsc{Ident}_{\mathbb{Q}}(2), \textsc{Ident}_{\mathbb{Q}}(3)$ についてはどうか？

部分的な結果として次が知られている．

事実 5.9. 以下が成り立つ．

- 単射な群の準同型 $\mathrm{FG}(\Sigma) \times \mathrm{FG}(\Sigma) \to \mathrm{GL}_3(\mathbb{Q})$ は存在しない (Ko and Niskanen and Potapov (2017) [21] を参考にすると証明できる)．したがって $\textsc{Ident}_{\mathbb{Z}}(3), \textsc{Ident}_{\mathbb{Q}}(3)$ の決定不能性を $\textsc{Ident}_{\mathbb{Z}}(4)$ の場合と同様の方法で示すことはできそうない．
- 入力を有理数成分の Heisenberg 群 $\mathrm{H}(3,\mathbb{Q}) = \left\{ \begin{pmatrix} 1 & a & c \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \middle| \, a,b,c \in \mathbb{Q} \right\}$ に限れば (多項式時間で) 決定可能 (Ko and Niskanen and Potapov (2017) [21])．
- $\textsc{Ident}_{\mathbb{Z}}(2)$ は NP 困難である (Bell and Potapov (2012) [22])．よって特に $\textsc{Ident}_{\mathbb{Z}}(3), \textsc{Ident}_{\mathbb{Q}}(3)$ も NP 困難である．

[*14] $\textsc{Ident}_{\mathbb{Z}}(n)$ というのは本稿だけの記号である．

5.2 その他の問題

上で述べたような問題以外にも様々な問題が未解決のまま残されている．

例えば Hilbert の第 10 問題は，有理数解の存否の決定可能性についてはまだ解決されていない．

未解決問題 5.10 (Hilbert の第 10 問題の変種). Hilbert の第 10 問題の $\mathbb{Z}$ を $\mathbb{Q}$ に変えた問題は決定可能か？

最後に，いくつかの決定不能問題を問題文だけ載せておく．問題の詳細や他の決定不能問題については例えば Poonen [12] などを参照してほしい．

事実 5.11. 以下の決定問題は全て決定不能である．

Wang tiling problem

Input: 各辺に色が塗られたタイル (単位正方形) の有限集合 S

Question: S の元を，隣接する辺の色が等しくなるように並べて全平面を充填できるか？ (同じタイルは何回使ってもよいが，回転させたり反転させたりすることはできないとする)

証明は新井 [6] などを参照のこと．

Polyomino tiling

Input: ポリオミノの有限集合 S (ポリオミノとは，有限個の単位正方形を辺で貼り合わせたような図形である．テトリスにおけるテトロミノのようなもの，と言えばわかりやすいだろうか．)

Question: S の元で全平面を充填できるか？ (同じポリオミノは何回使ってもよい．ポリオミノの回転や反転を許す場合と許さない場合の両方とも決定不能である．)

なお，入力されるポリオミノの個数を $|S| = 1$ に限った場合の決定可能性については未解決である．

群の有限表示

Input: 群 G の有限表示 (生成元とその関係式)

Question: G は自明な群/有限群/アーベル群/可解群/自由群か？ (全て決定不能)

群ではなく半群の場合の語の問題の決定不能性の証明が新井 [6] にある．

Homeomorphism problem

Input: 有限単体複体 M, N

Question: M と N は位相同型か？

6 終わりに

本稿の 4 節までの内容はほとんど Sipser [2] にある．Sipser [1, 2, 3] は計算の理論の標準的な教科書であり，本稿を読んで計算の理論に興味を持たれた方には一読を勧めたい．とても親切な教科書で行間などは一切なく，1 巻だけなら土日で読み切れてしまうと思う．3 巻には本稿で述べられなかった計算複雑性理論について書かれてあり，ミレニアム懸賞問題のひとつとして有名な P $\neq$ NP 予想についても丁寧な解説がある．

筆者が決定不能問題の魅力に取り憑かれてしまった最大のきっかけは B. Poonen のサーベイ論文 [12] を読んだことである．数学のありとあらゆる分野に決定不能問題が現れる様子を見るのは大変な衝撃であった．中でも特に Matrix mortality problem，Matrix identity problem は 2×2 行列や 3×3 行列といったとても素

朴な対象にまつわる問題であり，こんな一見簡単そうに見える問題が未だに解かれていないなんて信じられないと思ったことを覚えている．それ以来，これらの問題が決定可能なのか決定不能なのかが気になって論文を漁るようになり，また問題文もわかりやすいのでいろいろな所で人に話すようになった．読者にもこの驚きを少しでも伝えられたらと思って筆を執った次第である．

今回は計算のモデルとして Turing 機械を紹介したが，ラムダ計算，再帰関数，レジスター機械など，計算のモデルは他にもたくさんある．これらの計算モデルについて知りたい読者は新井 [6]，広瀬 [7]，高橋 [9, 10]，萩谷・西崎 [11] などの教科書を読んでみるとよいだろう．Hilbert の第 10 問題の決定不能性の証明については (筆者はまだ読めていないが) Matiyasevich 本人による教科書 [8] がある．次数の理論などのさらに進んだ内容については Soare [5] などの教科書を見てほしい．

本稿を読んで決定不能問題に興味を持つ人が少しでも増えることを願う．

参考文献

[1] M. Sipser (太田和夫・田中圭介 監訳, 阿部正幸・植田広樹・藤岡淳・渡辺治 訳), 計算理論の基礎 [原著第 2 版] 1. オートマトンと言語, 共立出版, 2008.

[2] M. Sipser (太田和夫・田中圭介 監訳, 阿部正幸・植田広樹・藤岡淳・渡辺治 訳), 計算理論の基礎 [原著第 2 版] 2. 計算可能性の理論, 共立出版, 2008.

[3] M. Sipser (太田和夫・田中圭介 監訳, 阿部正幸・植田広樹・藤岡淳・渡辺治 訳), 計算理論の基礎 [原著第 2 版] 3. 複雑さの理論, 共立出版, 2008.

[4] 河村彰星, はじめての計算可能性, 数学基礎論サマースクール 2017, `http://toshio-suzuki-logic.jp/meeting/summer2017.html`.

[5] R. I. Soare, *Turing Computability: Theory and Applications*, Springer, 2016.

[6] 新井敏康, 数学基礎論, 岩波書店, 2011.

[7] 広瀬健, 帰納的関数, 共立出版, 1989.

[8] Y. V. Matiyasevich, *Hilbert's Tenth Problem*, MIT Press, 1993.

[9] 高橋正子, 計算論 —計算可能性とラムダ計算—, 近代科学社, 1991.

[10] 高橋正子, コンピュータと数学, 朝倉書店, 2016.

[11] 萩谷昌己, 西崎真也, 論理と計算のしくみ, 岩波書店, 2007.

[12] B. Poonen, Undecidable problems: a sampler (2012), `https://arxiv.org/abs/1204.0299v2`.

[13] A. M. Turing, On computable numbers with an application to the Entscheidungsproblem, *Proc. London Math. Soc.* (2) **42** (1936) 230–265.

[14] E. L. Post, A variant of a recursively unsolvable problem, *Bull. Amer. Math. Soc.* **52** (1946) 264–268.

[15] M. S. Peterson, Unsolvability in 3×3 Matrices, *Stud. Appl. Math.* **49** (1970), 105–107.

[16] J. Cassaigne, T. Harju, J. Karhumaki, On the undecidability of freeness of matrix semigroups. *Int. J. Algebra Comput.* **09** no. 03n04 (1999) 295–305.

[17] O. Bournez, M. Branicky, The Mortality Problem for Matrices of Low Dimensions, *Theory Comput. Systems* **35** (2002) 433–448.

[18] P. C. Bell, M. Hirvensalo, I. Potapov, Mortality for 2×2 Matrices Is NP-Hard, in B. Rovan, V. Sassone, P. Widmayer (Eds.), *Mathematical Foundations of Computer Science 2012, Lecture Notes in Compute Science*, Vol. 7464, pp. 148–159, Springer Berlin Heidelberg, 2012.

[19] P. C. Bell, I. Potapov, The Identity Correspondence Problem and its Applications (2009), https://arxiv.org/abs/0902.1975v3.

[20] I. Potapov, P. Semukhin, Decidability of the Membership Problem for 2×2 integer matrices (2016), https://arxiv.org/abs/1604.02303v1.

[21] S. Ko, R. Niskanen, I. Potapov, On the Identity Problem for the Special Linear Group and the Heisenberg Group (2017), https://arxiv.org/abs/1706.04166v2.

[22] P. C. Bell, I. Potapov, On the Computational Complexity of Matrix Semigroup Problems, *Fundam. Inf.* **116** no. 1-4 (2012), 1–13.