Post の対応問題
Post Correspondence Problem

y.*

2018 年 2 月 1 日

最終更新日: 2018 年 2 月 1 日

概要

Post の対応問題とは，文字列の組の有限集合 $\{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$ が与えられたとき，$u_{i_1}u_{i_2}\cdots u_{i_k} = v_{i_1}v_{i_2}\cdots v_{i_k}$ となるような添字の有限列 $(i_1, i_2, \ldots, i_k)$ が存在するかどうかを判定する問題である．本稿では Post の対応問題が決定不能であることを示し，また Post の対応問題から派生した色々な決定問題を紹介する．前半は Sipser の教科書 [1] を参考にした．

Keywords: Post の対応問題 (Post correspondence problem).

目次

1 Post の対応問題 (PCP)

Post の対応問題とは，文字列の組の有限集合 $\{(u_1, v_1), (u_2, v_2), \ldots, (u_n, v_n)\}$ が入力されたとき「$u_{i_1}u_{i_2}\cdots u_{i_k} = v_{i_1}v_{i_2}\cdots v_{i_k}$ となるような添字の有限列 $(i_1, i_2, \ldots, i_k)$ は存在するか？」を判定する決定問題である．この問題を視覚的に捉えるために次のような定義を導入する．

* http://iso.2022.jp/

定義 1.1 (ドミノ，マッチ). 空文字列ではない[*1]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．1 つ以上のドミノからなる列

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

が**マッチ** (match) であるとは，上段を連結した文字列 $u_1u_2\cdots u_n$ と下段を連結した文字列 $v_1v_2\cdots v_n$ とが等しいことをいう．

この定義を用いると Post の対応問題は次のように述べることができる．

問題 1.2 (Post の対応問題 (Post correspondence problem; PCP)[*2]).
Input: ドミノの有限集合 P
Question: P はマッチを持つか？ (ただし，同じドミノは何回使ってもよい)

例 1.3.

$$P_1 := \left\{\left[\frac{\texttt{ab}}{\texttt{abab}}\right], \left[\frac{\texttt{b}}{\texttt{a}}\right], \left[\frac{\texttt{aba}}{\texttt{b}}\right], \left[\frac{\texttt{aa}}{\texttt{a}}\right]\right\}$$

とおくと，P_1 は次のマッチを持つ:

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

実際，水平方向に連結した文字列は上下段ともに ababababbaaaa となっている．

一方で

$$P_2 := \left\{\left[\frac{\texttt{abc}}{\texttt{ab}}\right], \left[\frac{\texttt{ca}}{\texttt{a}}\right], \left[\frac{\texttt{acc}}{\texttt{ba}}\right]\right\}$$

とおくと，P_2 はマッチを持たない．なぜなら，仮にマッチが存在したとすると，マッチの各段を水平方向に連結した 2 つの文字列の長さも等しくなければならないが，P_2 のどのドミノも下段より上段の方が長いためにそれは不可能だからである．

注意 1.4. Post の対応問題 1.2 では入力に使われる文字の種類 (アルファベット) は固定していない．これは (例 1.3 を見てもわかるように) どんな文字も入力に使うことができることを意味する．しかしながら，決定問題としてきちんと定式化するためには，あらかじめ固定されたアルファベット (これは有限でなければならない) を用いて全ての入力を表現する必要がある．「マッチが存在するかどうか」には実際の文字が何であるかは関係なく，同じ文字かどうかを区別することさえできれば文字の代わりに何を使ってもよい．したがって，文字の代わりに自然数を用いることで問題は解決できる．例えば入力が

$$\left\{\left[\frac{\texttt{abcdr}}{\texttt{abracadabra}}\right], \left[\frac{\texttt{吾輩は猫である}}{\texttt{名前はまだ無い}}\right]\right\}$$

のとき，それぞれの文字に自然数を割り当てることで実際には

$$\left\{\big((0,1,2,3,4),(0,1,4,0,2,0,3,0,1,4,0)\big), \big((5,6,7,8,9,10,11),(12,13,7,14,15,16,17)\big)\right\}$$

が入力されているのだと思えばよいわけである．

[*1] 空文字列ではないという仮定は本質的ではないが，証明を簡単にするためにこうしておく．
[*2] Post の原論文 [6] では correspondence decision problem と呼ばれている．

PCP の決定不能性を示すには停止問題 HALT の決定不能性を用いる．HALT の決定不能性の証明は「Turing 機械の定義と停止問題」[2] を参照のこと．

問題 1.5 (停止問題 (halting problem; HALT)).
Input: Turing 機械 M と文字列 w
Question: M は w を受理するか？

煩雑さを避けるため，HALT を直接 PCP に帰着するのではなく，次の決定問題を経由して証明する．

問題 1.6 (modified PCP; MPCP).
Input: ドミノの有限集合 P とその元 $d \in P$
Question: P は d を左端とするマッチを持つか？

補題 1.7. PCP が決定可能ならば MPCP も決定可能である．

証明. MPCP を判定するアルゴリズムを次のように構成する．MPCP の入力を

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

とする．$*, \lozenge$ を P に現れない新しい文字とする．一般に文字列 $w = w_1 \cdots w_l$ (各 w_i は 1 つの文字) に対し，

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

と定義する．このとき

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

とおくと[*3]

$$P \text{ が左端が } d \text{ であるマッチを持つ} \iff P' \text{ がマッチを持つ} \tag{1}$$

となることを示す．

($\Longrightarrow$) 明らか．

($\Longleftarrow$) P' の元で上段と下段の左端の文字が等しいものが $\left[\frac{\star u_1}{\star v_1 \star}\right]$ しかないため，マッチの左端は $\left[\frac{\star u_1}{\star v_1 \star}\right]$ でなければならない．よって，マッチを構成するドミノから $*$ と $\lozenge$ を全て取り除くことで d を左端とする P のマッチが得られる．

仮定より PCP を判定するアルゴリズムが存在するので，このアルゴリズムに P' を入力して計算させることで P' がマッチを持つかどうかがわかり，したがって (1) より P が d を左端とするマッチを持つかどうかがわかる． □

さて，いよいよ PCP の決定不能性を証明しよう．

定理 1.8 (Post [6], 1946). Post の対応問題 PCP は決定不能である．

[*3] $\lozenge$ は $\left[\frac{*\lozenge}{\lozenge}\right]$ の下段が空文字列にならないようにするためだけに入れてある．

証明. おおよそ Sipser の教科書 [1] に沿って証明する．仮に PCP が決定可能であるとすると，補題 1.7 より MPCP が決定可能である．このとき HALT も決定可能となってしまうことを示す．

HALT への入力を Turing 機械 $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ と入力文字列 $w = w_1 w_2 \cdots w_n$ $(w_i \in \Sigma)$ の組とする．ここで M は「ヘッドがテープの左端を見ている状態でさらにヘッドを左に動かそうとすることはない」という条件を満たすとしてよい (詳細は注意 1.11 を参照)．

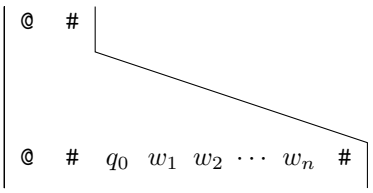
仮定より MPCP を判定するアルゴリズムがあるので，これを利用して M が w を受理するかどうかを調べたい．そのためには，ドミノの有限集合 P とその元 $d \in P$ を

$$M \text{ が } w \text{ を受理する} \iff P \text{ が } d \text{ を左端とするマッチを持つ} \tag{2}$$

を満たすように構成できることを示せば十分である．条件 (2) を満たすために，P のマッチから $M(w)$ の計算を復元できるようにしたい．そこで「d から始めてマッチを作ろうとすると，$M(w)$ の計算が勝手に進んでしまう」ように P を構成していく．これは言ってみれば“ドミノで Turing 機械の計算を模倣する”ということである．

@, #を Γ に含まれない新しい文字とする．$Q \cup \Gamma \cup \{@, \#\}$ 上の文字列を用いて P を以下のように構成する．

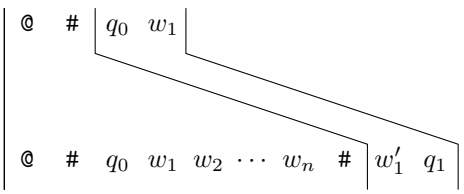
Step 1. まず，開始状況を表すドミノ $d = \left[\dfrac{@\#}{@\#q_0 w_1 \cdots w_n \#}\right]$ を P に追加する．ただし，w が空文字列 ε のときは $d = \left[\dfrac{@\#}{@\#q_0 \sqcup \#}\right]$ とする．



Step 2. d から始めてマッチを作るためには，上段の足りない部分を補う必要がある．このとき下段に 1 ステップ進んだ計算状況が現れるようにしたい．

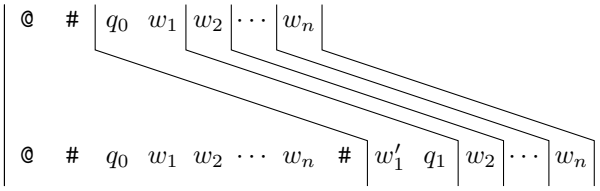
- $\delta(q, a) = (r, b, \mathrm{R})$ となる全ての $q, r \in Q \setminus \{q_{\text{reject}}\}, a, b \in \Gamma$ に対して $\left[\dfrac{qa}{br}\right]$ を P に追加する．
- $\delta(q, a) = (r, b, \mathrm{L})$ となる全ての $q, r \in Q \setminus \{q_{\text{reject}}\}, a, b \in \Gamma$ と $c \in \Gamma$ に対して $\left[\dfrac{cqa}{rcb}\right]$ を P に追加する．

例えば $\delta(q_0, w_1) = (q_1, w_1', \mathrm{R})$ のとき以下のようになる．

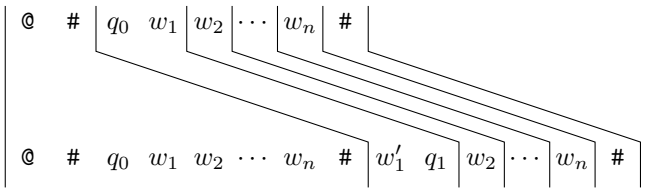


Step 3. ヘッドから遠い部分はテープの内容が変化しないので，上段の内容をそのまま下段にコピーすればよい．したがって，各 $a \in \Gamma$ に対して $\left[\dfrac{a}{a}\right]$ を P に加える．(これが MPCP を導入した理由である．PCP

だとこれ単体でマッチになってしまう．)

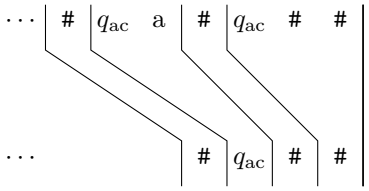


Step 4. $\left[\frac{\#}{\#}\right], \left[\frac{\#}{\lrcorner\#}\right]$ を P に追加する．(二番目のものは，計算の過程でヘッドが右に行きすぎたときにテープの余白を確保するために用いられる．)



Step 5. 状態が q_{accept} に達したら，マッチを完成させるために上段の内容を下段に“追い付かせる”必要がある．そのために各 $a \in \Gamma$ に対して $\left[\frac{aq_{\text{accept}}}{q_{\text{accept}}}\right], \left[\frac{q_{\text{accept}}a}{q_{\text{accept}}}\right]$ を P に加える．つまり，q_{accept} がテープ上の文字を“食べる”ことによって後始末を行うのである．

Step 6. テープ上の文字を全て掃除したら，最後に#の個数を合わせてマッチを完成させる．そのために $\left[\frac{q_{\text{accept}}\#\#}{\#}\right]$ を P に追加する．



最後に，構成した P が条件 (2) を満たすことを示そう．(直感的にはほとんど明らかだから，読み飛ばしても差し支えない．)

($\Longrightarrow$) P の作り方から明らか．

($\Longleftarrow$) P のマッチの各段を水平方向に連結した文字列 (上下段とも等しい) を $@\#C_0\#C_1\#\cdots\#C_{k-1}\#C_k$ とする．[*4]ここで，各 C_i は#を含まない (空かもしれない) 文字列である．このとき，各 $s \geq 0$ に対して「w に対する M の計算が 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 で表されており，しかも C_s の右隣の#を下段に含むドミノの上段の右端は C_s の左隣の#であること」[*5]を s に関する数学的帰納法で示す．まず $s = 0$ のとき，d の定義から C_0 は開始状況 $q_0w_1w_2\cdots w_n$ であり，C_0 の右隣の#を下段に含むドミノ (これは d である) の上段の右端は C_0 の左隣の#である．次に $s = s_0$ で主張が正しいとすると，帰納法の仮定から C_{s_0} の右隣の#を下段に含むドミノの上段が C_{s_0} の左隣の#なの

[*4] マッチの左端は d でなければならないので，左端は必ず@#である．

[*5] さらに細かいことを言えば $s < k$ であることも主張に含める必要がある (C_k の右隣の#は存在しないから)．

で，それより右側のドミノを全て捨てると

$$\begin{array}{ll}\text{(上段)} & \texttt{@\#}C_0\texttt{\#}C_1\texttt{\#}\cdots\texttt{\#}C_{s_0-1}\texttt{\#} \\ \text{(下段)} & \texttt{@\#}C_0\texttt{\#}C_1\texttt{\#}\cdots\texttt{\#}C_{s_0-1}\texttt{\#}C_{s_0}\texttt{\#}\end{array}$$

という形になっていなければならない (#を含むドミノは全て上下段とも右端の文字が#であることに注意する)．さらに帰納法の仮定から C_{s_0} は計算状況を表す文字列なので

$$C_{s_0} = a_{-l}a_{-l+1}\cdots a_{-1}qa_0a_1a_2\cdots a_r\,(a_i \in \Gamma, q \in Q, l, r \geq 0)$$

という形をしている．ここからもとのマッチを復元しよう．まず，$i \leq -2$ に対しては，先頭の 2 文字目までが Γ の元であるようなドミノは Step 3. で追加されたものしかないので，$\left[\frac{a_i}{a_i}\right]$ をそのまま並べるしかなく，C_{s_0+1} は $a_{-l}a_{-l+1}\cdots a_{-2}$ から始まることがわかる．また，上段が q だけのドミノは存在せず，上段に qa_0 が含まれるドミノは Step 2. で追加されたものしかないので，$\delta(q, a_0) = (r, b, \mathrm{L})$ のときは $\left[\frac{a_{-1}qa_0}{ra_{-1}b}\right]$ を並べるしかなく，$\delta(q, a_0) = (r, b, \mathrm{R})$ のときは $\left[\frac{a_{-1}}{a_{-1}}\right]\left[\frac{qa_0}{br}\right]$ を並べるしかない．$i \geq 1$ に対しては，上段が「左端が a_i で，その後ろに状態が続かない」となっているドミノは Step 3. で追加されたものしかないので，$\left[\frac{a_i}{a_i}\right]$ を並べるしかない．よって C_{s_0+1} はヘッドが左右どちらに動くかによって $a_{-l}\cdots a_{-2}ra_{-1}ba_1\cdots a_r$ または $a_{-l}\cdots a_{-2}a_{-1}bra_1\cdots a_r$ で始まる．最後に#を揃えるために，$\left[\frac{\texttt{\#}}{\texttt{\#}}\right], \left[\frac{\texttt{\#}}{\texttt{␣\#}}\right]$ のどちらを用いるかによって C_{s_0+1} の末尾に␣が付け加わるかどうかが変わる．(ここで d が使えないようにするためだけに@を付けてある．これは MPCP の定義を「d が左端のみに現れるようなマッチを持つか？」としておけば回避できる.[*6]) いずれにせよ C_{s_0+1} は C_{s_0} から 1 ステップ進んだ計算状況であり，また C_{s_0+1} の右隣の#を下段に含むドミノ ($\left[\frac{\texttt{\#}}{\texttt{\#}}\right]$ または $\left[\frac{\texttt{\#}}{\texttt{␣\#}}\right]$) の上段の右端は C_{s_0+1} の左隣の#になっている．以上より，(マッチが有限の長さであることから) $M(w)$ の計算は有限ステップで停止しなければならず，また P は q_{reject} を含まないのでその計算結果は受理でなければならない． □

注意 1.9. 上の証明で構成した P のマッチに一意性はない．いくつかのマッチを並べたものもまたマッチなので当たり前であるが，それ以外にも次のようなことからもわかる．

- q_{accept} がテープ上の文字を “食べる” 順番は自由である．
- $\left[\frac{\texttt{\#}}{\texttt{\#}}\right]$ の代わりに $\left[\frac{\texttt{\#}}{\texttt{␣\#}}\right]$ を用いてもよい (q_{accept} がテープ上の文字の掃除をするのにかかる回数が増える)．
- マッチの後ろに $\left[\frac{a}{a}\right]$ や $\left[\frac{q_{\mathrm{accept}}\texttt{␣}}{q_{\mathrm{accept}}}\right]\left[\frac{\texttt{\#}}{\texttt{␣\#}}\right]$ を付け加えてもマッチになる．

例 1.10. M を「Turing 機械の定義と停止問題」 [2] で定義した $\{\,\mathtt{a}^n\mathtt{b}^n \mid n \geq 0\,\}$ を判定する Turing 機械と

[*6] 補題 1.7 の証明において P' から $\left[\frac{\star u_1}{v_1\star}\right]$ を取り除けばよい．

する．HALT への入力を $(M, \mathtt{ab})$ とすると，

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

となり，P は例えば

$$\left[\frac{\mathtt{@\#}}{\mathtt{@\#}q_0\mathtt{ab\#}}\right]\left[\frac{q_0\mathtt{a}}{\mathtt{␣}q_1}\right]\left[\frac{\mathtt{b}}{\mathtt{b}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{q_1\mathtt{b}}{\mathtt{b}q_1}\right]\left[\frac{\mathtt{\#}}{\mathtt{␣\#}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{b}q_1\mathtt{␣}}{q_2\mathtt{b␣}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{\mathtt{␣}q_2\mathtt{b}}{q_3\mathtt{␣␣}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{q_3\mathtt{␣}}{\mathtt{␣}q_0}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]$$
$$\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{q_0\mathtt{␣}}{\mathtt{␣}q_{\mathrm{accept}}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{q_{\mathrm{accept}}\mathtt{␣}}{q_{\mathrm{accept}}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{\mathtt{␣}}{\mathtt{␣}}\right]\left[\frac{\mathtt{␣}q_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{\mathtt{␣}q_{\mathrm{accept}}}{q_{\mathrm{accept}}}\right]\left[\frac{\mathtt{\#}}{\mathtt{\#}}\right]\left[\frac{q_{\mathrm{accept}}\mathtt{\#\#}}{\mathtt{\#}}\right]$$

というマッチを持つ．このとき各段を水平方向に連結した文字列は上下段とも

$$\mathtt{@\#}q_0\mathtt{ab\#␣}q_1\mathtt{b\#␣b}q_1\mathtt{␣\#␣}q_2\mathtt{b␣\#}q_3\mathtt{␣␣␣\#␣}q_0\mathtt{␣␣\#␣␣}q_{\mathrm{accept}}\mathtt{␣\#␣␣}q_{\mathrm{accept}}\mathtt{\#␣}q_{\mathrm{accept}}\mathtt{\#}q_{\mathrm{accept}}\mathtt{\#\#}$$

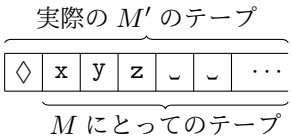
となり，入力 ab に対する M の受理計算履歴を含む．

注意 1.11. 任意に Turing 機械 $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\mathrm{accept}}, q_{\mathrm{reject}})$ が与えられたとき，M と同じ部分関数を計算する Turing 機械 M' を「ヘッドがテープの左端を見ている状態でさらにヘッドを左に動かそうとすることはない」という条件を満たすように構成することができる．基本的なアイデアはこうである: テープの左端を表す専用の記号 $\Diamond$ を用意しておき，$\Diamond$ を見たら問答無用で右に動くようにする．これで説明を終わってもよいのだが，ここではこのアイデアを正確に書き下してみることにしよう．Turing 機械 $M' = (Q', \Sigma, \Gamma', \delta', q_0', q_{\mathrm{accept}}, q_{\mathrm{reject}})$ を次のように定める．

$$\begin{aligned}
Q' &:= Q \cup \{\, \textsc{Write}(c) \mid c \in \Sigma \,\} \cup \{\textsc{GoToLeft}\},\\
\Gamma' &:= \Gamma \cup \{\Diamond\},\\
\delta'(q_0', a) &:= \begin{cases} (\textsc{Write}(a), \Diamond, \mathrm{R}) & (a \neq \mathtt{␣}),\\ (q_0, \Diamond, \mathrm{R}) & (a = \mathtt{␣}), \end{cases}\\
\delta'(\textsc{Write}(c), a) &:= \begin{cases} (\textsc{Write}(a), c, \mathrm{R}) & (a \neq \mathtt{␣}),\\ (\textsc{GoToLeft}, c, \mathrm{L}) & (a = \mathtt{␣}), \end{cases}\\
\delta'(\textsc{GoToLeft}, a) &:= \begin{cases} (\textsc{GoToLeft}, a, \mathrm{L}) & (a \neq \Diamond),\\ (q_0, \Diamond, \mathrm{R}) & (a = \Diamond), \end{cases}\\
\delta'(q, a) &:= \delta(q, a) \quad (q \in Q, a \in \Gamma),\\
\delta'(q, \Diamond) &:= (q, \Diamond, \mathrm{R}) \quad (q \in Q').
\end{aligned}$$

M' はまず左端のマスに $\Diamond$ を書き込み，入力文字列の全体ををを 1 マスぶんだけ右にずらす．そしてそれ以降は M の動作を模倣する．$\Diamond$ がテープの仮想的な左端の役割を果たすので，M にとっては $\Diamond$ の右隣のマスがテー

プの左端に「見えている」のである.[*7]



2 PCP の種々のバリエーション

本節では PCP から派生した様々な決定問題を紹介する.

2.1 入力を制限したもの

2.1.1 入力に使用できるアルファベットを制限したもの

PCP の入力に用いられる文字の種類に制限がないことを注意 1.4 で述べたが，この設定は本質的に必要というわけではない．実際，入力に使用可能な文字を $\{0,1\}$ に限っても決定不能なままである:

問題 2.1 (二進アルファベット上の PCP [1, 問題 5.18] (PCP over the binary alphabet)).
Input: 二進アルファベット $\{0,1\}$ 上の文字列からなるドミノの有限集合 P
Question: P はマッチを持つか？

定理 2.2. 二進アルファベット上の PCP は決定不能である.

証明. 二進アルファベット上の PCP が決定可能であると仮定して，PCP が決定可能となることを示す．PCP の入力 P をとる．P に現れる文字の全体を $\{a_0, a_1, \ldots, a_n\}$ とする．P に現れる $a_i\,(i = 0, 1, \ldots, n)$ を全て $10^i = 1\underbrace{0\cdots0}_{i}$ に置き換えたものを P' とすると,

$$P \text{ がマッチを持つ} \iff P' \text{ がマッチを持つ}$$

となる．(証明のアイデアは注意 1.4 とほとんど同じである．つまり，自然数を使う代わりに 0 の個数を使っただけである．) □

一方でアルファベットを $\{1\}$ に制限すると，上段と下段の長さが一致すればマッチになるので決定可能になる:

問題 2.3 (一進アルファベット上の PCP [1, 問題 5.17] (PCP over the unary alphabet)).
Input: 一進アルファベット $\{1\}$ 上の文字列からなるドミノの有限集合 P
Question: P はマッチを持つか？

定理 2.4. 一進アルファベット上の PCP は決定可能である.

証明の概要. 上段と下段の長さが等しいドミノがあれば受理する．そうでないとき，「上段の方が長いドミノ」と「下段の方が長いドミノ」の両方があれば受理し，そうでないとき拒否する． □

[*7] コンピューターに詳しい人は仮想機械のようなものと思うとわかりやすいのではないかと思う．M' がホストマシン，M がゲストマシンである.

2.1.2　入力の個数を制限したもの

入力のドミノを 2 個に限った場合は決定可能 [7]

入力を 5 個 (4 個でも？) に限っても決定不能

2.1.3　入力に含まれる文字列の長さを制限したもの

文字列の長さに関する制限については色々考えられるが，例えば次のような制限を加えると明らかに決定可能になってしまう:

問題 2.5 (silly PCP [1, 問題 5.19]).

Input: 上段と下段の長さが等しいようなドミノの有限集合 P

Question: P はマッチを持つか？

定理 2.6. silly PCP は決定可能である.

証明の概要. P がマッチを持つ $\iff$ P は上段と下段が一致しているドミノを含む. □

各段の文字列の長さを 2 以下に限っても決定不能 [8]

2.1.4　その他

marked PCP は決定可能 [9]

2.2　PCP の拡張

前小節までは入力を制限したものを扱ったが，ここでは入力や条件を変えた問題を紹介する.

2.2.1　マッチの条件を緩めたもの

circular PCP は決定不能 [10]

n-Permutation PCP は決定不能 [11]

2.2.2　入力を変更したもの

Identity correspondence problem (ICP) は決定不能 [12]

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] 2. 計算可能性の理論, 共立出版, 2008.

[2] y., Turing 機械の定義と停止問題, http://iso.2022.jp/math/undecidable-problems/files/turing-machine-and-the-halting-problem.pdf, 2018.

[3] 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://www.graco.c.u-tokyo.ac.jp/~kawamura/t/summer_H29/, 2017.

[4] y., 決定不能問題の話, 第 10 回すうがく徒のつどい, http://iso.2022.jp/math/tsudoi/10/slide.pdf, 2017.

[5] y., 決定不能問題から始める計算可能性理論入門, 都数 12 月総会, http://iso.2022.jp/math/tosuu-2017-12/resume.pdf, 2017.

[6] E. L. Post, A variant of a recursively unsolvable problem, *Bull. Amer. Math. Soc.* **52** (1946) 264–268, https://doi.org/10.1090/S0002-9904-1946-08555-9.

[7] A. Ehrenfeucht, J. Karhumäki, G. Rozenberg, The (generalized) post correspondence problem with lists consisting of two words is decidable, *Theoret. Comput. Sci.* **21** no. 2 (1982) 119–144, https://doi.org/10.1016/0304-3975(89)90080-7.

[8] V. Halava, T. Harju, M. Hirvensalo, J. Karhumäki, Post Correspondence Problem for short words, *Inform. Process. Lett.* **108** no. 3 (2008) 115–118, https://doi.org/10.1016/J.IPL.2008.04.013.

[9] V. Halava, M. Hirvensalo, R. de Wolf, Marked PCP is decidable, *Theoret. Comput. Sci.* **255** no. 1–2 (2001) 193–204, https://doi.org/10.1016/S0304-3975(99)00163-2.

[10] K. Ruohonen, On some variants of Post's Correspondence Problem, *Acta Inform.* **19** no. 4 (1983) 357–367, https://doi.org/10.1007/BF00290732.

[11] M. Ernvall, V. Halava, T. Harju, On the n-permutation Post Correspondence Problem, *Theoret. Comput. Sci.* **601** (2015) 15–20, https://doi.org/10.1016/J.TCS.2015.07.021.

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

変更履歴

2018/02/01　2 節を [1] の演習問題のみの状態で公開