

チューリング機械の領域計算量の厳密な階層について

九州芸術工科大学 岩本 宙造 (Chuzo Iwamoto)
九州大学・工学部 岩間 一雄 (Kazuo Iwama)

1 はじめに

領域計算量を論じる際には、作業テープの記号数を固定するか否かで二つの場合が考えられる。記号数を固定しない場合には、 $\inf_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$ なる関数 $g(n), f(n)$ に対し、 $f(n)$ 領域の決定性 TM は $g(n)$ 領域の決定性 TM より受理能力が真に高いことが知られている [1, 2]。線形加速定理が成立するので [1, 2]、この階層は稠密であると言える。一方、テープ記号を m に固定した場合には、それほど稠密な結果は知られていなかった。最も良い結果として、如何なる整数 $r \geq 1$ 、定数 $b > a \geq 0$ に対して、 m 記号の bn^r 領域決定性 TM は、 m 記号の an^r 領域決定性 TM より真に受理能力が高いことが知られていた [4]。(これは、如何なる $\epsilon > 0$ に対しても、 $(1+\epsilon)an^r$ 領域は an^r 領域より受理能力が高いことを意味する。) テープ記号数を固定した場合には、 $f(n)$ 領域の決定性 TM は $(f(n)-c)$ 領域の決定性 TM で模倣できることが知られているに過ぎないことから、[4] の階層には改善の余地が十分残されており、それほど厳密な階層とは言えない。

本稿では、作業テープの記号数を m に固定して、より厳密な領域計算量の階層を示す。 $s(n)$ を任意の強構成可能関数とする。このとき、如何なる整数 m 、如何に小さい定数 $\epsilon, \epsilon_1, \epsilon_2 > 0$ に対しても、以下の条件を満たす言語 $L \subseteq \{0, 1\}^*$ が存在する。(i) m 記号の決定性 TM T で $\left(s(n) + \frac{s(n)}{\sqrt{(3-\epsilon)n}} + (4+\epsilon_1)\log_m s(n) + (3+\epsilon_2)\log_m n\right)$ 領域で受理できる。(ii) m 記号の如何なる決定性 TM T_x でも $s(n)$ 領域では L を受理できない。(iii) T は必ず停止するが、 T_x は無限ループに入ることを入力を受理しないこともある。 $s(n)$ が $\log n$ より真に速く成長する関数とすれば、 m 記号の $(1+o(1))s(n)$ 領域決定性 TM は、 m 記号の $s(n)$ 領域決定性 TM より受理能力が高いと言える。つまり、[4] における定数係数 $1+\epsilon$ は、「1 に漸近する関数」に置き換えられたことになる。さらに、[4] では $s(n)$ は an^r と書ける関数と制限されていたが、我々の結果は、如何なる強構成可能関数 $s(n)$ に対しても成立する。

証明は、これまでに知られる階層の証明と同様に、対角線論法による。しかし、我々の結果は、これまでの文献にはなかった二つの新しいアイデアに基づいている。その一つは、入力テープの使い方である。我々のモデルは、いわゆるオフライン TM であり、二方向にヘッドを動かせる入力テープと作業テープからなる。階層定理

の対角線論法による証明において重要な点は、 T の入力テープ上に符号化列として与えられた TM T_x を、 T で模倣する部分である。その際、 T_x の符号化列の後ろに、十分長い挿入列 (padding sequence) を付け加えた列を、 T の入力列として与えることで、 T が T_x を模倣するのに十分な領域が確保されるようになる。そのため、挿入列は十分な長さがあればどのような列でも良く、 1^l といった任意の列が使われてきた。本稿では、ある複雑な構造を持つ列を挿入列として選び、その構造を利用することで、より厳密な階層定理を証明する。直観的に言えば、それまでの挿入列では、入力列を利用して一つのカウンタを模倣できていたが、我々の新しい挿入列は、二つのカウンタを模倣することが可能となる。

もう一つのアイデアは、Sipser によって提案された手法の新たな応用である。階層定理の証明におけるもう一つの重要な点は、 T_x が無限ループに入っているか否かを T が如何に判定するかである。最も単純な方法は、 T_x の動作数を数え上げることであるが、 T_x は $s(n)$ 領域なので、 T は $s(n)$ 程度の余分な領域を必要とする。つまり、新たな $s(n)$ 領域をカウンタに使用し、残りの $s(n)$ 領域を T_x の作業テープの模倣に用いることになり、 T が $2s(n)$ 領域 TM になることは避けられそうにない。Sipser は、[7] において、受理計算状況から深さ優先で初期計算状況を探る手法を提案している。この手法を使うことで、ごく少ない領域の増加で T_x がループに入るか否かを T は判定できるようになる。

2 これまでの結果とモデル

モデルは、いわゆる決定性オフライン チューリング機械 M であり、有限制御部、左右端に特別の終端記号の書かれた読みとり専用入力テープ、一本の作業テープから構成されている。ただし、作業テープに関しては、文献によって以下の点で定義が微妙に異なる。(i) 一方向に無限か、二方向に無限か、(ii) 作業テープ上に空白記号 B を書くことができるか否か、(iii) 一方向無限であれば、テープの他方の端には特別の終端記号が置かれているか否か。

テープ記号が制限されていない場合について階層定理を論じる場合は、これらの違いは問題は生じない。しかし、作業テープが一方向無限で、テープ記号は $0, 1, B$ のみ、左端の終端記号は使えないと仮定すれば、次のような問題が生じる。明らかに、 M が最初にすべきこ

とは、特別な文字列（例えば、111）を作業テープの左端に書いて、ヘッドが作業テープから落ちないようにすることである。しかしながら、この方法では、左端の111を見分けるために、作業テープ上の他の場所に文字列111が現れないようにしなくてはならない。そうすると、必要な領域量が定数係数の範囲で増加してしまうことは避けられそうにない。もし、両方向に無限の作業テープを用いることが可能なら、空白記号 B を終端記号の代わりに使えるので、この問題は生じない。

テープ記号が制限された場合の階層定理に関しては、Ibarra [3] の結果が知られている。そのモデルは、一つの作業テープと一つの作業テープヘッドをもつ。Ibarra は、(i) 一方向無限と二方向無限の両方の場合を考えており、(ii) 作業テープのヘッドは空白記号 B を書けると仮定しているが、(iii) 作業テープのヘッドが左端を認識できるとの仮定は使っていない。文献 [3] において、 $m+1$ 記号で n^r 領域の TM で受理できるが、 m 記号で n^r 領域の如何なる TM でも受理できない言語が存在することが示されている。この階層定理は、決定性と非決定性の両方の場合に成立する。しかし、作業テープが二方向無限で、かつ、決定性の場合、テープ記号数 $m+1$ は $m+2$ に増加する。

Ibarra と Sahni [4] は、[3] と同じモデルを用いて、一方向無限の作業テープの場合について、上記の階層を改善した。文献 [4] において、如何なる整数 $r \geq 1$ 、如何なる定数 $b > a \geq 0$ に対して、 m 記号で bn^r 領域の TM で受理できるが、 m 記号で an^r 領域の如何なる TM でも受理できない言語が存在することが示されている。

Seiferas [5, 6] は、領域やテープ記号、作業テープのヘッド数を制限することで、クラスを分離する多くの結果を示した。Seiferas の TM は、一つの作業テープ上に $l \geq 1$ 個のヘッドを持つ。Seiferas は、(i) 作業テープは一方向無限、(ii) 作業テープのヘッドは空白記号 B を書くことが可能で、(iii) 作業テープのヘッドは左端を認識できると仮定している。文献 [5] において、 $s_0(n) - 2s(n) - l \log_m s(n) - \log_m n \neq O(1)$ なる関数 $s_0(n), s(n)$ に対して、 m 記号で $s_0(n)$ 領域、 $(l+2)$ ヘッドの決定性 TM で受理できるが、 m 記号で $s(n)$ 領域、 l ヘッドの如何なる決定性 TM でも受理できない言語が存在することが示されている。（この階層性は、作業テープのヘッド数を増加させてクラスの違いを示しているが、領域量に対数を加えることで作業テープのヘッド数を減らすことができることが知られている。つまり、作業テープの記号数が m に固定されている場合、 $l+k$ 個のヘッドを持つ $s(n)$ 領域決定性 TM は、 $(k$ 個の新たなカウンタを使うことで) l 個のヘッドを持つ TM で $s(n) + (k+1+\epsilon) \log_m s(n)$ 領域で模倣可能である。しかし、この模倣は $l=1$ の場合は成立

しない。）

3 結果

本稿の TM は、読みとり専用の入力テープ一本と読み書きできる作業テープ一本からなるオフライン TM である。入力テープの両端には、特別の終端記号が置かれている。本稿の TM は、入力テープと作業テープにそれぞれ一つのヘッドをもつ。また、(i) 作業テープは二方向に無限で、(ii) 作業テープのヘッドは、空白記号 B を書けないと仮定する。（従って、作業テープのヘッドが訪れたマスには必ず非空白記号が書き込まれている。）

最初、入力ヘッドは入力テープの左端のマスに置かれている。作業テープの記号は、 $0, 1, \dots, m-1$ とする。最初、作業テープの各マスには空白記号 B が書かれている。初期状態、受理状態、非受理状態は、それぞれ唯一であり、 q_1, q_2, q_3 で表される。TM は、無限ループに入ることで入力列を受理しないこともある。全ての n に対して、如何なる入力を与えられても作業テープ上でちょうど $s(n)$ マス使用する TM が存在するとき、 $s(n)$ を強構成可能関数という。

定理 1. $s(n)$ を任意の強構成可能関数とする。このとき、如何なる整数 $m \geq 2$ 、如何に小さい定数 $\epsilon, \epsilon_1, \epsilon_2 > 0$ に対しても、以下の条件を満たす言語 $L \subseteq \{0, 1\}^*$ が存在する。(i) m 記号の決定性 TM T で $\left(s(n) + \frac{s(n)}{\sqrt{(\frac{2}{3}-\epsilon)n}} + (4 + \epsilon_1) \log_m s(n) + (3 + \epsilon_2) \log_m n\right)$ 領域で受理できる。(ii) m 記号の如何なる決定性 TM T_x でも $s(n)$ 領域では受理できない。(iii) T は必ず停止するが、 T_x は停止するとは限らない。（証明は節 4 で与える。）

系 1. 強構成可能関数 $s(n)$ が、 $s(n) \neq O(\log n)$ を満たせば、 m 記号で $(1 + o(1))s(n)$ 領域の決定性 TM は、 m 記号の $s(n)$ 領域の決定性 TM より受理能力が高い。

注意 1. $s(n) \leq \sqrt{(\frac{2}{3}-\epsilon)n}$ のときは、定理 1 の $\frac{s(n)}{\sqrt{(\frac{2}{3}-\epsilon)n}} + \log_m n$ の項を除くことができる。（証明に幾つかの修正が必要である。詳しくは、節 4 で述べる。）故に、 $s(n)$ が $\log n$ の多項式のときは、領域 $s(n)$ に $(2 + \epsilon_2) \log_m n$ を加えるだけで、 m 記号の領域限定決定性 TM で受理できる言語のクラスは真に大きくなることが分かる。

強構成可能関数には、整数 $c \geq 1$ 、有理数 $k \geq 1, t > 1$ に対し、 $n!, 2^n, n^k, \log^t n, c$ などがある。また、 $s_1(n)$ と $s_2(n)$ が強構成可能であれば、 $s_1(n) + s_2(n), s_1(n)s_2(n), 2^{s_1(n)}$ などとも強構成可能である。

本稿では、二方向無限の作業テープを仮定し、空白記号 B は書けないものとしている。この仮定は、定理 1

が全ての強構成可能関数に対して成立するという目的のために導入されている。しかし、 $n!, 2^n, n^k, \log^t n, c$ など、良い振舞いをする具体的な関数や、これら特定の関数から構成される関数 $s_1(n) + s_2(n), s_1(n)s_2(n), 2^{s_1(n)}$ に対しては、定理 1 は [4] と全く同じモデル上で成立する。

注 意 2. $s(n)$ を、 $n!, 2^n, n^k, \log^t n, c$ と いった良い振舞いをする具体的な関数、または、良い振舞いをする関数から構成される関数 $s_1(n) + s_2(n), s_1(n)s_2(n), 2^{s_1(n)}$ とする。このとき、定理 1 は、一方向無限の作業テープをもち、左端に特別の記号を持たない決定性 TM で成立する。さらに、作業テープに空白記号 B を書けるとし、テープ記号を $B, 0$ のみに限定しても定理 1 は成立する。

故に、我々の結果は、全く同じモデル上で [4] の結果を改善するとともに、領域関数 $s(n)$ を一般化したことになる。[5, 6] のように特別の終端記号が作業テープの一端に置かれているとすると、我々の階層定理は一方向無限の決定性 TM 上で全ての強構成可能関数 $s(n)$ に対して成立する。

4 定理 1 の証明

如何なる $s(n)$ 領域 TM T_x でも受理できない言語 $L \subseteq \{0, 1\}^*$ を、 $\left(s(n) + \frac{s(n)}{\sqrt{(\frac{2}{3} - \epsilon)n}} + (4 + \epsilon_1) \log_m s(n) + (3 + \epsilon_2) \log_m n\right)$ 領域で受理する TM で、必ず停止する T を作成する。まず最初に、TM の符号化規則を定める。以下では、 $s(n) \geq \sqrt{(\frac{2}{3} - \epsilon)n}$ と仮定する。

$(s(n) < \sqrt{(\frac{2}{3} - \epsilon)n})$ の場合は、幾つかの修正が必要であり、必要に応じて以下の各節で説明する。この修正により、定理 1 から $\frac{s(n)}{\sqrt{(\frac{2}{3} - \epsilon)n}} + \log_m n$ の項を除くことができる。

4.1 TM の符号化規則

TM の状態を $q_1, q_2, \dots$ で表す。状態 q_i は、長さ $i + 1$ の文字列 10^i に符号化され、作業テープの記号 $B, 0, 1, 2, \dots$ は、文字列 $10, 100, 1000, 10000, \dots$ に符号化される。文字列 10 (100) はヘッドが左 (右) 動くことを意味する。例えば、遷移関数 $\delta_1(q_5, B, 2) = (q_3, 1, L, R)$ は、以下の文字列に符号化される。

$$\underbrace{100000}_{q_5} \underbrace{10}_B \underbrace{10000}_{2} \underbrace{1000}_{q_3} \underbrace{1000}_{1} \underbrace{10}_L \underbrace{100}_R$$

TM の符号化は、遷移関数の符号化を連結してできる文字列の後ろに初期状態、受理状態、非受理状態を表す文字列を連結したものである。さらに、この TM の符号

化の後ろには、任意に長い文字列が連結されている。TM の符号化の部分符号化部 (encoding part)、その後ろの任意に長い文字列を挿入部 (padding part) と呼ぶ。挿入部は、以下の構造を持つ (図 1 も参照)。

$$\underbrace{110^b 10^1}_{\leq 3b+2} \underbrace{110^b 10^2}_{\leq 3b+2} \dots \underbrace{110^b 10^h}_{\leq 3b+2} \dots \underbrace{110^b 10^b}_{\leq 3b+2} 1100 \dots 0 \quad (1)$$

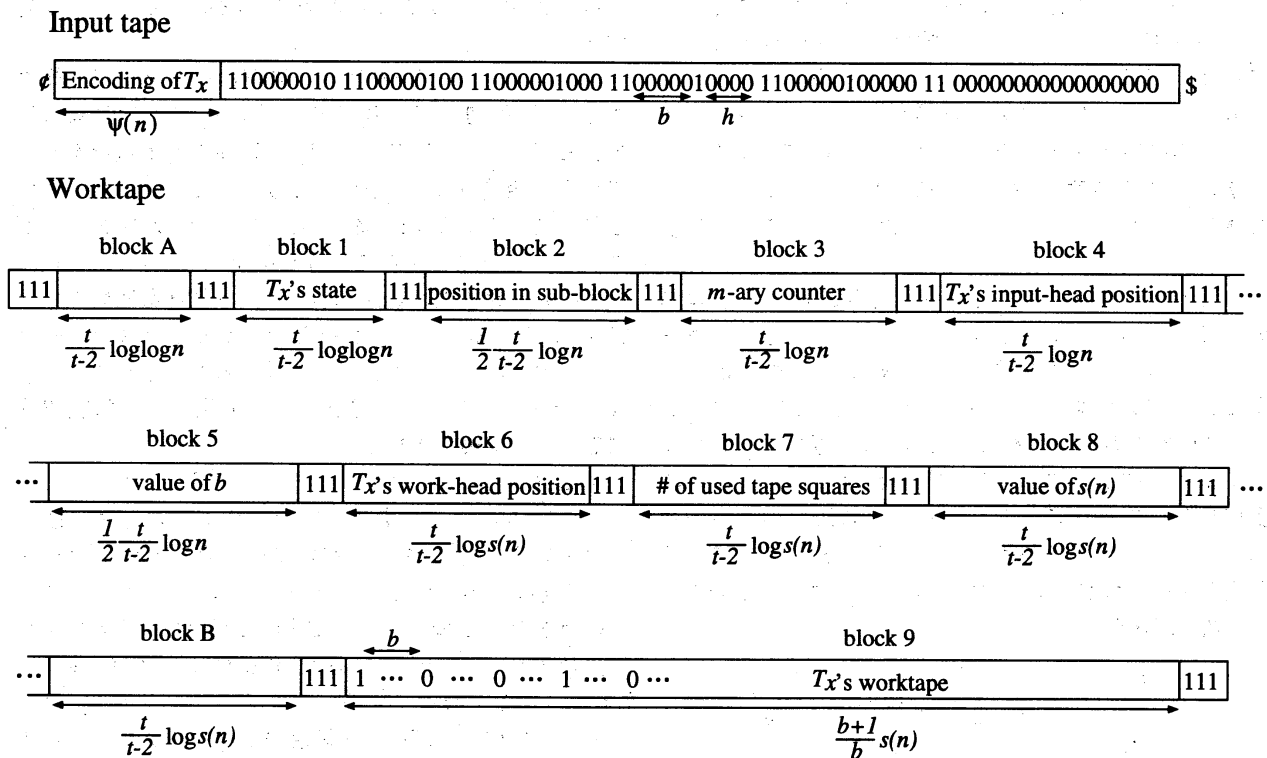
h 番目の部分列は、長さ $b + 3$ の文字列 $110^b 1$ と長さ h の文字列 $00 \dots 0$ からなる。一番最後の部分列は $110^b 10^b$ であり、その後ろには長さが高々 $3b + 2$ の文字列 $00 \dots 0$ が連結されている。挿入部の最後の文字列 $00 \dots 0$ の長さが $3b + 2$ より大きければ、 $b + 1$ を改めて b とおいて上記と同じ構造が作れてしまうことに注意が必要である。

如何に成長が遅い関数 $\psi(n) \neq O(1)$ に対しても、十分長い挿入部が符号化部に連結されていれば、入力列の先頭から $\psi(n)$ 番目までに符号化部が含まれるようになる。そのような十分長い挿入部には、如何なる小さい $\epsilon > 0$ に対しても $b \geq \sqrt{(\frac{2}{3} - \epsilon)n}$ を満たす部分列 $110^b 10^b$ が含まれることになる。

T は、その入力列 x を、TM T_x の符号化部と挿入部の連結とみなし、 T が x を受理するのは、 T_x が x を受理しないときに限るとする。言い換えれば、 T がその入力 x を受理しないのは、以下の条件が一つでも満たされない場合である。(i) x が文法的に間違っている。(ii) x の符号化部がいずれかの決定性 TM に対応しない。(iii) x の挿入部の構造が正しくない。

4.2 T_x の計算状況の表現

T は、作業テープを 11 個のブロックに分ける。これらは、ブロック 1, 2, ..., 9, ブロック A, B と呼ばれる (図 1 参照)。二つのブロック間の境界は長さ t の文字列 1^t である。ただし、 $t \geq 3$ は十分大きい定数である。(t の値は、後で与える不等式 (2) が成立するように大きくとる。) ブロック 1, 4, 6 は、それぞれ T_x の状態、入力ヘッド位置、作業テープのヘッド位置を表すために用いる。 T_x は、 m 個の記号が使える。そこで、ブロック 3 は m 進カウンタ、ブロック 7 は T_x の作業テープ上で使用したマス数を m 進数の文字列として蓄える。さらに、ブロック 5, 8 は b と $s(n)$ の値を m 進数の文字列としてそれぞれ蓄える。(ブロック 3, 5, 7, 8 の使用目的は後で述べる。) T_x の作業テープはブロック 9 で表されるが、extra bit と呼ばれる記号が b 文字ごとに等間隔で挿入されている (図 1 参照)。extra bit は 0 または 1 である。(b の値は、入力部の挿入部の b と同じであることに注意。) 二つの extra bit に挟まれた長さ b の領域をサブブロックと呼ぶ。extra bit は、本証明で重要な役割を演じる。この点については後で述べる。各 extra bit は、(i) それに引き続きサブブ

Figure 1: Input tape and worktape of T

ロックに T_x の作業テープのヘッドが存在するか否か、(ii) どれが最初のサブブロックで、どれが最後のサブブロックかを表す。図1では、4番目のサブブロックの直前の extra bit が1となっている。これは、 T_x の作業テープのヘッドが4番目のサブブロックのあるマスに置かれていることを意味する。最初と最後を除く残り全ての extra bit は0である。($s(n) < b$ のときは、ブロック9には extra bit は挿入されない。ブロック9は長さ $s(n) < b$ の一つのサブブロックから構成されていると考える。) T_x の作業テープのヘッドが、あるサブブロックの中の h 番目のマスに置いてあるとき、ブロック2には h の値が蓄えられている。

ブロック9には、extra bit が b 文字ごとに等間隔で挿入されているので、長さは $\frac{b+1}{b} s(n)$ となる。(b の値は、入力列の挿入部に与えられている。) ($s(n) < b$ のときは、ブロック9には extra bit は挿入されないで、長さは $s(n)$ のままである。また、ブロック5も不要となる。さらに、ブロック2はブロック6で代用できる。したがって、定理1から $\frac{s(n)}{\sqrt{(\frac{2}{3}-\epsilon)n}} + \log_m n$ の項を除くことができる。) もし、ブロックの中に文字列 1^t が現れるとすると、 T はその文字列 1^t と境界を表す文字列 1^t の区別がつかなくなる。そこで、長さ2の文字列 00 を $t-2$ マスごとに等間隔で挿入する。故に、ブロックA, 1, ブロック2, 5, ブロック3, 4, ブロック6, 7, 8, Bの長さは、それぞれ、 $\frac{t}{t-2} \log_m \log_m n$, $\frac{1}{2} \frac{t}{t-2} \log_m n$, $\frac{t}{t-2} \log_m n$, $\frac{t}{t-2} \log_m s(n)$ となる。(こ

うすることで、挿入された空間 00 の上に必要に応じて目印 01 を置くことが可能となる。以下では、二つのブロックに含まれる値が等しいかを判定する場面が頻繁に出てくる。そのような場合、長さ $t-2$ の各文字列を対応する同じ長さの文字列と比較していく。その際、今どの文字列とどの文字列を比較しているのかを示すために、二つの目印 01 を用いる。)

$s(n)$ は、強構成可能関数であるので、まず T は作業テープ上でちょうど $s(n)$ マス使う TM を模倣する。 T は、全ての非空白記号を0に変え、その両端に境界を表す 1^t を生成する。(0が書かれた部分がブロック9になるが、extra bit は後で挿入する。) これらの0の数を m 進カウンタで数え上げることにより、 T はブロックBを生成することができる。ブロック6, 7, 8も同様である。ブロック3, 4は、入力列 x の長さを数え上げることにより生成できる。ブロック2, 5は、入力列 x の長さを一文字置きに数え上げることにより生成できる。また、ブロック1, Aはブロック3の長さ $\log_m n$ を数え上げることにより生成できる。今のところ、ブロック9は $s(n)$ マスしか含んでいない。そこで、以下の方法で記号0を b 文字ごとに等間隔で挿入することで長さを $\frac{b+1}{b} s(n)$ にする。(i) T は、入力ヘッドを長さ b の文字列 00...0の最初の文字の上におく。(ii) T は、その作業ヘッドをブロック9の最初の文字の上に置く。 T の入力ヘッドが h 個の0を順に読んでいく作業と並行して、 T の作業テープのヘッドを右へ動かしていく。

T の入力ヘッドが記号1に着いたとき、 T の作業テープのヘッドはブロック9の**b**番目のマスに着いたことになる。(iii)そこで、一つの記号1が挿入される。つまり、ブロック9の右端の境界を右に一マスシフトする。この手続きを繰り返すことで、 T はブロック9に含まれるマスの数を $\frac{b+1}{b}s(n)$ に増やすことができる。最後に、ブロック9に含まれる文字1の全てを0に変える。

ひとたび上記11ブロックが生成されたら、 T は境界記号1[†]を決して消したり、移動したりしない。さらに、 T はブロック9に含まれるextra bitの位置も変えない。 T の作業テープのヘッドは、常に11ブロックの中で作業を行なう。もし、 T が T_x を模倣するのにより多くの領域が必要であることが判明したら、 T は非受理状態で停止する。したがって、如何に小さい定数 $\varepsilon, \varepsilon_1, \varepsilon_2 > 0$ が与えられても、大きな n に対して、 T が使うマスの数が以下の式で押えられるような十分大きい定数 t が存在する。

$$\begin{aligned} & \frac{1+b}{b}s(n) + \frac{4t}{t-2}\log_m s(n) + \frac{3t}{t-2}\log_m n \\ & + \frac{2t}{t-2}\log_m \log_m n + 12t \\ \leq & s(n) + \frac{s(n)}{\sqrt{(\frac{2}{3}-\varepsilon)n}} + (4+\varepsilon_1)\log_m s(n) \\ & + (3+\varepsilon_2)\log_m n \end{aligned} \quad (2)$$

4.3 T_x の動作の模倣

まず、 T は次の条件を満たすTM (T_x とする)が存在するかを判定する。(i)入力列 x の符号化部が T_x の正しい符号化になっている。(ii) T_x は決定性である。(iii)符号化部は(1)で示した構造を持つ。(i)は、有限オートマトンのように単純に前から読んでいくだけで判定できる。(ii) T は、まず T_x の符号化をブロック1にコピーする。(入力列 x の符号化部が十分長ければ、入力列の先頭から $\psi(n) \leq \log_m \log_m n$ までに x の符号化部が含まれる。そうでなければ(ブロック1に入りきらなければ) T は非受理状態で停止する。)入力ヘッドと作業テープのヘッドを用いて、遷移関数の全ての組を比較することで、 T は T_x が決定性か否かを判定する。(iii) T は、挿入部に含まれる11なる文字列の後ろに同じ長さの文字列00...0を従えているかを調べる。この00...0の長さを**b**とおく。 m 進カウンタを用いて、符号化部の最後の0の数が高々 $3b+2$ 個であることを確かめる。各**b**に対して、 T は二つの隣接する部分列中の 10^b と 10^{b+1} をブロック3の**m**進カウンタを使って比較する。さらに、 T は最後の部分列が 110^b10^b であることを確かめる。もし、 x が条件(i),(ii),(iii)を満たさない場合は、 T は非受理状態で停止する。以下

では、入力列 x はこれらの条件を満たしていると仮定する。

T_x の1ステップの動作を、以下の7ステップで模倣する。(i)最初、 T の作業テープは T_x の初期計算状況を含んでいる。(ii) T は、 T_x の現在の計算状況を含んでいると仮定する。(iii) T は T_x の入力ヘッドが今何を読んでいるかを調べる。(iv) T は T_x の作業テープのヘッドが今何を読んでいるかを調べる。(v) T は T_x の現在の状態を調べる。(vi) T は、状態遷移関数の中で T_x の状態、入力ヘッドが読んでいる文字、作業テープのヘッドが読んでいる文字が一致するものを見つける。(vii) T は T_x の計算状況を、(vi)で見つけた状態遷移関数に従って遷移させる。

(i)最初、 T_x の作業テープのヘッドは最初マスに置かれているので(つまり、最初のサブブロックの最初マスの上に置かれているので)、最初のサブブロックの直前のextra bitは1となっている(サブブロックの境界は1となる)。(最初と最後のサブブロックは、他のサブブロックと区別するため、直前のextra bitが常に1となっている。したがって、もし T_x の作業テープのヘッドが最初か最後のサブブロックにあるときは、 T はそのことを状態に記憶しておく。)ブロック9の残りのマスは、全て0である。また、 T_x の作業テープのヘッドは、サブブロックの最初マスに置いてあるので、ブロック2は値1を**m**進数として蓄える。ブロック1は T_x の初期状態 q_1 を蓄える。ブロック3は、**m**進カウンタであり、最初は値0が入っている。入力ヘッドは最初入力テープの最初マスに置かれているので、ブロック4は値1を含んでいる。ブロック6,7は値0が蓄えられている。ブロック5,8は、それぞれ常に値**b**, $s(n)$ が蓄えられている。厳密に言えば、ブロック9の全てのマスは最初は空白記号**B**でなくてはならない。その代わりに、ブロック9の全てのマスには**B**の代わりに0が入っている。そこで、ブロック6,7を使った特別な手続きで、 T_x の作業テープのヘッドが**B**を読んでいるか否かを判定するようにしている。(この手続きに関しては後述する。)

(ii)ブロック1~9は、 T_x の現在の計算状況を表しているとする。

(iii)ブロック4に含まれる値を*i*とおく。 T の作業テープのヘッドは、値0をブロック3に**m**進数として書く。また、 T の入力ヘッドは入力列 x の左端の記号の上に置く。(a) T の入力ヘッドを右方向へ一マス動かし、(b) T の作業テープのヘッドはブロック3の値を1だけ増加させる。ブロック3,4に含まれる値が一致するまで、(a),(b)の手続きを繰り返すことにより、 T の入力ヘッドを入力列 x の*i*番目の記号の上に動かすことができる。

(iv)ブロック9では、最初は全ての記号が0であっ

たことから分かるように、ブロック 9 には、 T_x の作業テープが B を読んでいるか否かに関する情報は何も含まれていない。そこで、 B を読んでいるか否かについては、ブロック 6,7 を使って判断する。もしブロック 6,7 の値が等しい、または、もしブロック 6 の値が -1 ならば、 T は T_x の作業テープのヘッドは空白記号 B を現在読んでいると判断できる。ブロック 6,7 はこの目的のためのみに用いる。(T_x の作業テープが読んでいる文字を見つける際には、 T はブロック 2, extra bit, 入力列 x の挿入部を使う。)

T_x の作業ヘッドは今 B を読んでいないとする。仮定よりブロック 9,6 は、それぞれ T_x の作業テープ、 T_x の作業テープのヘッド位置を蓄えている。ブロック 6 に蓄えられている値を j とおく。まず、以下のような単純な手続きを考えてみる。(a) (iii) と同様の方法で、 T の入力ヘッドを入力列 x の j 番目の文字に置く。(つまり、 T は入力ヘッドを右に動かしていくのと並行して、ブロック 3 の値をブロック 5 の値に一致するまで 0 から一つずつ大きくしていく。)(b) T の作業テープをブロック 9 の最初マスへ動かす。(c) T の入力ヘッドを j 番目のマスから左へ動かすのと並行して、 T の作業ヘッドをブロック 9 の最初マスから右へ動かしていく。 T の入力ヘッドが入力テープの左の終端記号に着いたとき、 T の作業テープはブロック 9 の j 番目のセルに着く。(つまり、入力列を一進カウンタとして使う。)しかしながら、この方法では、次の二つの重大な問題が生じる。(1) $j \leq n$ の場合しか成り立たない。(2) 入力ヘッドは、今、入力テープの左端のマスに置かれているので、 T の入力テープは j の値を再びカウントすることができない。したがって、この方法では T の作業テープのヘッドは、ブロック 9 の最初マスに戻ることができない。(もし、作業テープが一方向無限で、しかも、特別な終端記号が作業テープの左端に置かれているとすれば、 T の作業テープは作業テープの左端にヘッドを移動してから、ブロック 9 の先頭にヘッドを移動できる。二方向無限で空白記号 B が書けないとする我々のモデルの場合を考える。もし、空白記号 B を終端記号の代わりに用いたとすれば、その B は非空白記号に変えられるので、使用するマス数が増加してしまう。)そこで、入力列の挿入部をうまく利用する。

仮定より、(a) ブロック 9 は T_x の現在の作業テープの内容を蓄えており、(b) ブロック 9 では b 文字ごとに等間隔で extra bit が挿入され(各 b 文字はサブブロックと呼ばれる)、(c) サブブロックの直前の extra bit が 1 であるのは、それが最初または最後のサブブロックであるとき、または、そのサブブロックに T_x の作業テープのヘッドが存在するときであり、(d) T_x の作業テープのヘッドのサブブロックの中での位置はブロック 2 に蓄えられている。

入力列の挿入部は、(1) に示す構造となっており、 110^b なる部分列で h 番目のものは、直後に 10^h が続いている(図 1 も参照)。ブロック 2 に含まれる値を h とおく。(iii) と同様に、 $(h$ 個の 11 を通り越して) T の入力ヘッドを h 番目の部分列 $110^b 10^h$ へ持っていく。ここでは、前半部分 0^b と後半部分 0^h を、それぞれサイズ b, h の一進カウンタとして用いる。これら二つをそれぞれカウンタ U_b, U_h と呼ぶ。まず、手続きの概略を示しておく。(a) T はカウンタ U_h を見つける。(b) カウンタ U_h に隣接するカウンタ U_b を使って、 T の作業テープのヘッドを extra bit 1 へ移動させる。(extra bit 1 は、その直後のサブブロックに T_x の作業テープヘッドが含まれることを表す。)(c) カウンタ U_h を使って、そのサブブロック中の h 番目のマスに T の作業ヘッドを動かす。 $(s(n) < b$ のときは、block 9 は長さ $s(n) < b$ のサブブロック一つから構成される。したがって、手続きのうち U_b を使う部分は不要である。)

T の作業テープのヘッドは、最初のサブブロックの直前の extra bit の上に置かれているとする。 T の入力ヘッドを U_b 上を左から右へ動かすのと並行して、 T の作業テープのヘッドを右へ動かす。こうすることで、 T の作業テープのヘッドは 2 番目の extra bit にたどりつく。次に、 T の入力ヘッドを U_b 上を右から左へ動かすのと並行して、 T の作業テープのヘッドを右へ動かす。こうして、 T の作業テープのヘッドは 3 番目の extra bit にたどりつく。 T の作業テープのヘッドが extra bit 1 と出会うまで、この手続きを繰り返す。(extra bit 1 と出会えば、その直後のサブブロックには T_x の作業テープのヘッドが含まれる。)

次に、 T の入力ヘッドをカウンタ U_h 上で左から右へ動かすのと並行して、 T の作業テープのヘッドを右へ動かす。こうすることで、 T の作業テープのヘッドは、そのサブブロックの h 番目のマスにたどりつく。こうして、 T は、 T_x が今何を読んでいるかを知ることができる。そして、 T の入力ヘッドを U_h 上を左から右へ動かすのと並行して、 T の作業テープのヘッドを左へ動かす。こうして、 T の作業テープのヘッドを extra bit の上に戻すことができる。上記と同様に、カウンタ U_b を繰り返し用いることにより、 T の作業テープのヘッドを最初の extra bit の上に戻すことができる。

(最初の extra bit は 1 なので、他の extra bit とは見わけがつく。)

(v) T_x の現在の状態はブロック 1 に蓄えられている。 T の作業テープのヘッドを単にブロック 1 に移動させるだけでよい。

(vi) T の入力ヘッドは、入力列 x の符号化部の状態遷移関数を順に見ていき、 T_x の現在の計算状況に一致する状態遷移関数 (δ_1 とする) を見つけ出す(後述の

図3参照)。

(vii) T はブロック1に蓄えられた T_x の状態を遷移関数 δ_1 に従って変化させる。もし T_x の入力ヘッドが右(左)に動くとき、 T はブロック4に蓄えられた値に1を加える(値から1を引く)。ここで、三つの場合が考えられる。(a) ブロック6,7に蓄えられた値が等しければ(T_x の作業テープのヘッドが右端の非空白記号の隣の B を読んでいれば)、または、ブロック6の値が-1のときは(T_x の作業テープのヘッドが左端の非空白記号の隣の B を読んでいれば)、ブロック7の値を1だけ増加させる。ブロック7の値がブロック8の値より大きければ、 T は非受理状態で停止する。そうでなければ(b)を実行する。(c) (iv)と同様にして、 T の作業テープのヘッドはextra bitが1であるサブブロックの h 番目のマスにアクセスして δ_1 に従って記号を書き換える。(ただし、ブロック6の値が $h = -1$ だったときは後で述べる処理をする。)

T_x の作業ヘッドが右(左)へ動くときは、 T はブロック2の値を1だけ大きく(小さく)する。もし、ブロック2の値が b で、かつ、 T_x の作業テープのヘッドが右へ動く時はブロック2の値を0にし、1が立っていたextra bitを0に変え、その次に現れるextra bitを1にする。もし、ブロック2の値が1で、かつ、 T_x の作業テープのヘッドが左へ動く時はブロック2の値を0にし、1が立っていたextra bitを0に変え、その前のextra bitを1にする。

もしブロック6の値が $h = -1$ であった(つまり、今は-2または0に変えられている)場合は、ブロック6の値に1を加え、extra bitを除くブロック9の内容を全体に渡って右に1マスだけシフトする。(extra bitの位置は T は絶対に動かさないことに注意。)そうするために、 T の入力ヘッドを最初の U_b の先頭に持っていく。 T の作業テープがブロック9の内容をシフトしている間、 T の作業テープのサブブロック内でのヘッド位置を、 U_b 内での入力ヘッド位置で模倣する。こうすることで T の作業テープはextra bitを認識することができる。また、最初と最後のextra bitは1が立っているのでブロック9から T の作業テープのヘッドが飛び出すことはない。シフトによって空けられた最初のマス(最初のサブブロックの最初のマス)に、 δ_1 に従って記号を書き加える。

4.4 T_x が x を受理するか否かの判定

いま、 T_x が x を受理しないときに限り、 T は入力列 x を受理するという T を作成している。ここで注意すべきこととして、 T_x は無限ループに入ることによって x を受理しないこともあり得るということである。もし、 T が T_x を単純に初期状況から最終状況に向けて模倣していくと、 T は T_x がループに入っているのか否かが認識

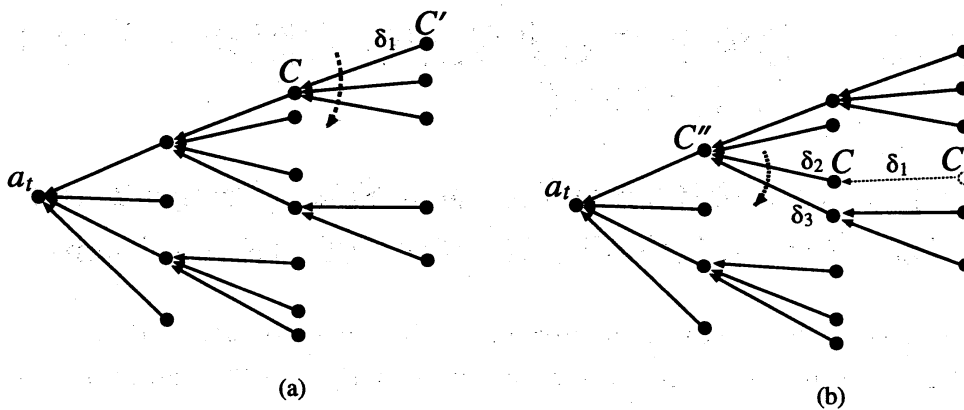
できなくなる。(つまり、 T_x が無限ループに入って x を受理しないときは、 T も無限ループに入って x を受理しなくなる。) Sipser [7] は、 T_x の動作数を数え上げることにし、この判定が可能であることを示した。ここでは、その手法を適用する。

以下では、TM T_x が入力を受理するときは、全ての非空白記号を0に変え、左端の0の隣の B にヘッドを戻し、唯一の受理状態で停止すると仮定する。(この仮定で、問題は何も生じないということを補題1で述べる。) こうすることで、 $s(n)$ 領域限定における異なる受理計算状況の数は $s(n)$ となる(つまり、最終状況における作業テープの使用量の違いだけである)。作業テープ上に t 個の0を含む受理計算状況を a_t で表す。ただし、 $t = 1, 2, 3, \dots, s(n)$ とする。

各節点が $s(n)$ 領域限定の計算状況を表し、枝が計算状況の間の遷移を表した有限の有向グラフを考える。 T_x は決定性であるので、このグラフの出次数は高々1である。つまり、 a_t は木の根になるということである(図2参照)。 T_x が入力列 x を受理するのは、 T_x の初期状況が、ある受理状況 a_t を根とする木に属している場合に限られる。 $t = 1, 2, \dots, s(n)$ の各値に対して、 T は a_t を根とする木に深さ優先探索を実行する。

以下の記述は、[7]を元に行っている。各 t に対して a_t を根とする木の全ての節点を訪れるためには、 T は、(i) t の値、(ii) T_x の現在の計算状況 C 、(iii) 1ステップ前の T_x の計算状況 (C' とする) を現在の計算状況 C に遷移させることの可能な遷移関数 δ_t を憶えておかなくてはならない。ブロック B の長さは、 $\frac{t}{t-2} \log_m s(n)$ であるので(図1参照)、 T は t の値をブロック B に蓄える。 T_x の現在の計算状況 C は、ブロック1~9で表現する。挿入部が十分長い入力文字列 x を考えれば、如何に成長の遅い関数 $\psi(n) \neq O(1)$ に対しても、先頭から $\psi(n)$ 番目までに符号化部が含まれることになる。そのような入力 x に対しては、遷移関数 δ_t は長さ $\log_m \log_m n$ のブロック A に蓄えることができる。もし、ブロック6に蓄えられた値が、以下の手続き中に $s(n)$ の値より大きくなれば、 T は非受理状態で停止する。

(a) T_x の現在の計算状況を C とおく(図2(a)参照)。節4.3と同様にして、 T_x の作業テープのヘッドが今読んでいるマスと左右のマスに書かれた記号、 T_x の入力ヘッドが今読んでいるマスと左右のマスに書かれた記号、 T_x の今の状態を T は知ることができる。 T_x の符号化は遷移関数の符号化を連結したものであり、それらは入力列 x の符号化部に含まれている(図3参照)。そこで、 T_x の遷移関数を左から右へ順に見ていき、1ステップの遷移で計算状況 C' を C に遷移できる遷移関数 (δ_1 とする) が存在するかを調べる。もし、そのような δ_1 が見つければ、 T_x の遷移関数を C' に戻

Figure 2: Depth-first search of the tree rooted at some a_t

す (図 2(a)). もし, C' が初期計算状況でなければ, T は再び手続き (a) を C' に対して適用する.

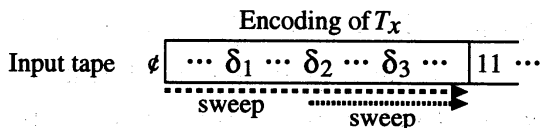


Figure 3: Sweep of the next move functions

(b) もし, そのような δ_1 が存在しない場合は (図 2(b) 参照), 節 4.3 と同様に T は単純に T_x の動作を 1 ステップだけ模倣する. 1 ステップ後の計算状況を C'' とする. C を C'' に遷移させる遷移関数を δ_2 とする. 次に, C'' に対して手続き (a) を適用するが, 遷移関数を順に見ていく際には δ_2 から右の遷移関数のみを順に見ていく (図 2(b), 3 参照).

T は $t = 1, 2, \dots, s(n)$ の全てに対して, a_t を根とする木の全節点を訪れる. もし, ある a_t を根とする木の中に T_x の初期計算状況が見つかった場合は, T は非受理状態で停止する. もし, T_x の初期計算状況が, $t = 1, 2, \dots, s(n)$ の如何なる t に対しても a_t を根とする木に含まれないときは, T は入力列 x を受理する. 次の補題で本定理の証明を終る.

補題 1. m 記号で $s(n)$ 領域の如何なる TM でも, 上記のように定義した TM T で受理される言語を受理することはできない.

証明. T で受理される言語を受理できる m 記号, $s(n)$ 領域の TM (T_x とする) が存在すると仮定する. すると, 次の条件 (i), (ii) を満足する TM T'_x が存在する. (i) T'_x は T_x と同じ言語を受理する. (ii) T'_x は入力を受理する際には, 全ての非空白記号を 0 に変え, 左端の 0 の隣の空白記号 B の上にヘッドを戻し, 唯一の受理状態で停止する. TM が全ての非空白記号を 0 に変えるとき, 作業テープのヘッドは非空白記号の左右端に隣接する二つの空白記号 B を 0 に変えることも考えられる. 故に, T_x は $s(n)+2$ 領域 TM である. 如何な

る $s(n)+2$ 領域 TM T'_x に対しても, T'_x と同じ言語を受理する $s(n)$ TM T''_x が存在する [4]. 従って, 上記条件を満たし, T と同じ言語を受理する $s(n)$ 領域 TM T''_x が存在する.

T''_x の符号化に十分長い挿入部を連結した入力列 x を考える. この x を T に入力列として与えたら, T が使うマスの数は不等式 (2) で押えられる. 故に, 上記で述べた模倣が問題なく進み, T が x を受理するのは, T''_x が受理しないときに限られる. これは, 仮定に矛盾する. $\square$

参考文献

- [1] J. HARTMANIS, P.M. LEWIS II and R.E. STEARNS, Hierarchies of memory limited computations, in: *Proc. 6th IEEE Symp. on Switching Circuit Theory and Logical Design*, pp. 179-190, 1973.
- [2] J.E. Hopcroft and J.D. Ullman, Introduction to automata theory, languages and computation, Addison-Wesley, Reading, MA, 1979.
- [3] O.H. Ibarra, A hierarchy theorem for polynomial-space recognition, *SIAM J. Comput.* 3 3 (1974) 184-187.
- [4] O.H. Ibarra and S.K. Sahni, Hierarchies of Turing machines with restricted tape alphabet size, *JCSS* 11 (1975) 56-67.
- [5] J.I. Seiferas, Technique for separating space complexity classes, *JCSS* 14 (1977) 73-99.
- [6] J.I. Seiferas, Technique for separating space complexity classes, *JCSS* 14 (1977) 100-129.
- [7] M. Sipser, Halting space-bounded computations, *TCS* 10 (1980) 335-338.