計算による実数の分類

新井 敏康（神戸大学工学研究科）

11/1/2007

ここでは断らない限り、数は自然数を、集合は自然数の集合を、関数は自然数上の部分関数、つまり定義域が $\mathbb{N}^n$ の部分集合である関数を意味する。

1 計算可能性

ここでいう 計算 は、紙に書けるような有限的な対象を入力として、ある固定された有限個の規則（アルゴリズム とか プログラム と呼ぶ）に則って、入力に対して機械的な操作を有限回施して出力を得るような過程である。

次のことを仮定する：

1. 出力は各入力に対して高々ひとつ。
2. 入力の引数はアルゴリズムごとに決まっている。
3. 計算の過程は離散的でひとつずつの計算ステップに分かれる。
4. 計算の各ステップは、入力を決めれば一意に機械的に定まっている。（決定性）
5. とくに計算を始める状態と終わる、停止する状態は予めアルゴリムによって定められている。

上で述べた「計算」の概念はまだあまりにも漠としており数学的な定義ではない。しかしこれに見合う 計算可能性(computability) の正確な定義はいくつも提出され、しかもそれらはすべて同値であることが知られている。

先ず、入出力と計算の途中経過を記録する対象はすべて紙に書けるような有限的対象としたが、これらは本質的にいくつかの文字・記号から成る有限列と考えてよいだろう。するとそのような対象は自然数で表現できる。これを コード化（符号化 (en)coding）という。

1.1 コード化

コード化の例を見る。初めに長さに制限のない自然数の有限列 $(x_0, \ldots, x_{n-1})\,(n \geq 0)$ をひとつの自然数 $\langle x_0, \ldots, x_{n-1} \rangle$ でコードする。

例えば p_n を n 番目の素数 $(p_0 = 2, p_1 = 3, \ldots)$ として

$$\langle x_0, \ldots, x_{n-1} \rangle := \prod_{i<n} p_i^{1+x_i} = p_0^{1+x_0} \cdots p_{n-1}^{1+x_{n-1}}$$

と定めればよい。特に空列（$n = 0$）のコードは $\langle\rangle = 1$ である。このとき $exp(x, i) = \max\{y \leq x : p_i^y | x\}$（$x$ を割り切る p_i の最大ベキ）[1]とすると

$$n \dot{-} 1 := \begin{cases} n-1 & n > 0 \\ 0 & n = 0 \end{cases}$$

について、復号化(decoding) $(x)_i$ と列（のコード）x の長さ(length) $lh(\langle x_0, \ldots, x_{n-1} \rangle) = n$ は

$$\begin{aligned} (x)_i &= exp(x, i) \dot{-} 1 \\ lh(x) &= \min\{i \leq x : exp(x, i) = 0\} \end{aligned}$$

で与えられ、また列のコードの集合 Seq は

$$x \in Seq \Leftrightarrow \forall i \leq x (p_i | x \to i < lh(x))$$

また

$$x \neq 0 \to (x)_i < x \wedge lh(x) < x \tag{1}$$

ともなっている。

一般に記号 $a_0, \cdots, a_N$ から成る文字列 $x_0 \cdots x_k$ をコード化するには、先ず、各記号のコード $\lceil a_i \rceil$ を定め、（例えば、a_i のコードを $\lceil a_i \rceil = p_i$）文字列 $x_0 \cdots x_k$ のコードを $\langle \lceil x_0 \rceil, \ldots, \lceil x_k \rceil \rangle$ とすればよい。

これで計算に関わるプログラム、入出力、計算過程はすべて自然数でコードできるので自然数であると思ってよいことになる。

以下で全域的でない関数（部分関数(partial function)）を扱うので、少し記法を定めておく。

値を持たない、定義されていない(undefined) かもしれない式 e, f について、$e \downarrow$ は e が定義されていることを表し、$e \simeq f$ は、両辺が定義されていてそれらの値が等しいか、もしくは両辺ともに定義されていないことを表す。

自然数上の関係 $R(n)$ について、$\mu n.R(n)$ で、$R(n)$ を満たす自然数が存在すればそのような最小のものを表し、そうでなければ $\mu n.R(n)$ は定義されない。

関係 $R \subseteq \mathbb{N}^k$ が 計算可能 とは、その特徴関数

$$\chi_R(\vec{x}) = \begin{cases} 1 & \text{if } R(\vec{x}) \\ 0 & \text{otherwise} \end{cases}$$

が計算可能なときにいう。

[1] $exp(0, i) = 0$ と定めておく。

1.2 基本的な諸結果

上記の計算に対する理解のもとに次のことは正しいとしてよいだろう（「計算」の定義を正確に与えてないのだからいまのところ証明はできないが)。

1. 関係「列 $y = \langle y_0, \ldots, y_k \rangle$ は、引数 n のプログラム e に入力 $\vec{x} = (x_0, \ldots, x_{n-1})$ を与えた環境での（k ステップまでの）計算過程（のコード）である」は計算可能である。これを

$$Comp(e, \vec{x}, y)$$

と書く。

ここで各 y_i は計算過程中の計算状態を表し、y_{i+1} はその次の計算状態である。

2. 各 n について

$$T_n(e, \vec{x}, y)$$

で、「引数 n のプログラム e について $Comp(e, \vec{x}, y)$ かつ $y = \langle y_0, \ldots, y_k \rangle$ の最後 y_k の計算状態は計算の終了状態である」を表すとする。これも計算可能である。

計算過程は一意的なので、$T_n(e, \vec{x}, y)$ を満たす y は $e, \vec{x}$ ごとに高々ひとつである。この y （が存在すればそれ）を $e, \vec{x}$ の 計算数(computation number) と呼ぶ。

3. 計算数 y について計算結果（出力）を $U(y)$ で表す。これも計算可能である。
4.

$$\{e\}(\vec{x}) \simeq U(\mu y.T_n(e, \vec{x}, y))$$

と定義する。

計算可能な（＝プログラムがある）n-変数部分関数 f について、e（そのプログラムのコード！）が存在して

$$f(x_0, \ldots, x_{n-1}) \simeq \{e\}(x_0, \ldots, x_{n-1}) \simeq U(\mu y.T_n(e, x_0, \ldots, x_{n-1}, y))$$

となる (normal form thm)。

5. 計算数 s 以下の計算のみ考えて

$$\{e\}_s(\vec{x}) \simeq U(\mu y[y \leq s \,\&\, T_n(e, \vec{x}, y)])$$

とおく。すると明らかに

$$\{e\}(\vec{x}) \simeq z \Leftrightarrow \exists s[\{e\}_s(\vec{x}) \simeq z].$$

2　半計算可能集合

集合 A が計算可能であるとは、$n \in ?A$ という問いに Yes/No で（正しく）答えてくれるアルゴリズムがあるということである。これに対して、問い $n \in ?A$ が正しい場合、そしてその場合に限って Yes と答え、$n \notin A$ の場合には如何なる答えも出さない（計算が終了しない、定義されない）ようなアルゴリズムが存在するとき A は、半計算可能(semicomputable) と呼ばれる。

つまり、計算可能関数 $\{e\}$ の定義域を

$$W_e := \{\vec{x} : \{e\}(\vec{x}) \downarrow\}$$

と書き、計算可能関数の定義域になっている関係 $R(\vec{x})$

$$\exists e \forall \vec{x}[R(\vec{x}) \leftrightarrow \vec{x} \in W_e]$$

が半計算可能ということになる。

補題 2.1 関係 $R(\vec{x})$ が半計算可能であるのは、ある計算可能関係 S により

$$\forall \vec{x}[R(\vec{x}) \leftrightarrow \exists y\, S(\vec{x}, y)]$$

と表せるときである。

補題 2.2 $K := \{x : x \in W_x\} = \{x : \{x\}(x) \downarrow\}$ は半計算可能だがその補集合は半計算可能でない。特に K は計算可能ではない。

しかもつぎの意味で、半計算可能集合の中で 完全*(RE-complete)* である：各半計算可能集合 X について、計算可能関数 F で

$$x \in X \Leftrightarrow F(x) \in K$$

となるものが取れる。

証明. K が半計算可能なのは $x \in K \leftrightarrow \exists y\, T_1(x, x, y)$ なので補題 2.1 よりよい。

K の補集合が半計算可能としてその指標 e を取る：

$$\forall x[x \notin K \leftrightarrow x \in W_e]$$

ここで $x = e$ と取ると

$$e \notin W_e \leftrightarrow e \notin K \leftrightarrow e \in W_e$$

となり矛盾。よって K は計算可能ではない。 □

3　相対化

集合 A を 神託(oracle) として用いる計算（A-計算 もしくは A に相対化 された計算）の概念を導入する。これはプログラムの命令リストの中に

$$n \in ?A$$

という新しい種類のものを加えることにより得られる。

つまり、A-計算の途中で得られた値 n について、$n \in A$ かどうか訊いてその答え (Yes/No) を得てそれを後の計算に利用できる。

このような命令を含んだプログラムを A-プログラムと呼ぶ。A-計算で計算できる（部分）関数を A-計算可能関数という。

「A-プログラム e に入力 $\vec{x}$ を与えて計算を実行して計算が終了するまでの A-計算過程が y である」を表す $T_n^A(e,\vec{x},y)$ は、A-計算可能で

$$\{e\}^A(\vec{x}) \simeq U(\mu y.T_n^A(e,\vec{x},y)).$$

ところで、終了する各計算過程は有限ステップで終了するので、その間に神託に伺いをたてるのも有限回である：計算数 y について、この計算では神託 A の内で高々 $0 \in ?A, \ldots, y-1 \in ?A$ という問いしか発生していない。つまりこの計算は

$$A|y := \{i < y : i \in A\}$$

にしか依存しない。

いま集合 A の y までの情報を記録した数を

$$\bar{A}(y) := \sum_{i<y} 2^i a_i$$

ここで

$$a_i := \begin{cases} 1 & i \in A \\ 0 & i \notin A \end{cases}$$

として $\bar{A}(y)$ を $A|y$ と同一視する。上で述べたことは A-計算では有限情報 $\bar{A}(y)$ で足りているということである。

従ってある（相対化されない）計算可能述語 $T_{n,1}$ が存在して

$$T_n^A(e,\vec{x},y) \leftrightarrow T_{n,1}(e,\vec{x},y,\bar{A}(y))$$

となる。

相対化されていない場合と同様に計算数を限定して

$$\{e\}_s^A(\vec{x}) \simeq U(\mu y[y \le s \,\&\, T_n^A(e,\vec{x},y)]) \simeq U(\mu y[y \le s \,\&\, T_{n,1}(e,\vec{x},y,\bar{A}(y))])$$

とおく。

すると明らかに

$$\begin{aligned} &\{e\}^A(\vec{x}) \simeq y \Rightarrow \exists s[\{e\}_s^{\bar{A}(s)}(\vec{x}) \simeq y \,\& \\ &\max\{e,\vec{x},y\} < s \,\&\, \forall B \forall t > s(\bar{B}(s) = \bar{A}(s) \Rightarrow \{e\}_t^B(\vec{x}) \simeq y)] \end{aligned} \tag{2}$$

4　計算により比較不可能な実数の構成

集合 A, B について、A をその特徴関数 χ_A と同一視して、χ_A が B-計算可能なとき、A は B に Turing 還元可能(Turing reducible) といって

$$A \leq_T B$$

と書く。これは次と同値である：

$$\forall x[(x \in A \to \{e\}^B(x) \simeq 1) \wedge (x \notin A \to \{e\}^B(x) \simeq 0)] \tag{3}$$

また

$$A <_T B :\Leftrightarrow A \leq_T B \,\&\, B \not\leq_T A$$

同値関係

$$A \equiv_T B :\Leftrightarrow A \leq_T B \leq_T A$$

で定め、この同値関係による同値類を 次数(degree) と呼ぶ。次数上にも関係 $\leq_T$ 及び $<_T$ は自然に拡張され、次数上では $\leq_T$ は半順序になる。

次数の理論は、$\boldsymbol{D} = P(\mathbb{N})/\equiv_T$ 上での半順序 $\leq_T$ の構造を調べることを目的としている。集合 A の次数、すなわち A の属する $\equiv_T$ に関する同値類を $[A]$ と書こう。

$\mathbf{0}$ で計算可能集合の次数を表し、K の次数を $\mathbf{0}'$(jump) と書く。補題 2.2 よりどんな半計算可能集合 A についても

$$\mathbf{0} \leq_T [A] \leq_T \mathbf{0}'.$$

E. Post(1944) は、それまでに得られていた半計算可能集合が計算可能かもしくは RE-完全な K と T-同値なものしかなかった、つまりそれらの次数でいえば $\mathbf{0}$ か $\mathbf{0}'$ しかなかった状況から次の問題をたてた：

Post の問題：　計算可能でないが半計算可能な集合 A で
$A <_T K$ となるものは存在するか？

これに答えたのが R. M. Frieberg(1957) と A. A. Muchnik(1956)（独立になされた）で、その手法は 優先論法(priority argument) と呼ばれ、後に様々に洗練されて次数の理論の主要武器としてその理論を深めた。

Frieberg と Muchnik は単に Post の問題に答えるだけでなく半順序 $\leq_T$ が線形順序でないことまで示している。

定理 4.1 *(Friedberg-Muchnik の定理)*
半計算可能集合 A, B で半順序 $\leq_T$ に関して比較不能なものがつくれる。従ってその次数は $\mathbf{0} <_T [A], [B] <_T \mathbf{0}'$ である。

半計算可能集合 A と B でどんな e についても次の条件 R_{2e} と R_{2e+1} を同時に満たすものをつくりたい：

$$\begin{aligned} R_{2e} &: A \neq \{e\}^B \\ R_{2e+1} &: B \neq \{e\}^A \end{aligned}$$

これは (3) によれば各 e ごとに数 x_{2e} をつくり（x_{2e+1} も同様）

$$[x_{2e} \in A \,\&\, \{e\}^B(x_{2e}) \not\simeq 1] \vee [x_{2e} \notin A \,\&\, \{e\}^B(x_{2e}) \not\simeq 0] \tag{4}$$

としなければならない。

さて証明のアイデアであるが、先ず集合 A, B の構成および B-計算 $\{e\}^B(n)$ と A-計算 $\{e\}^A(n)$ をすべて時系列に並べて考える。時刻 s は計算のステップ数にほぼ対応しているが、むしろ小節 1.2 で述べた計算数 ($\{e\}^B(n) \simeq U(\mu s.T_{1,1}(e, n, s, \bar{B}(s)))$) と思えばよい。集合 A, B の時刻 s までの部分情報を含む集合をそれぞれ A_s, B_s と書くとこれらは

1. A_s, B_s は有限集合。
2. $\{A_s\}_s$, $\{B_s\}_s$ はともに増加列 ($s < t \Rightarrow A_s \subseteq A_t$) で $A = \bigcup_s A_s, B = \bigcup_s B_s$.

条件 R_{2e} つまり (4) を満たすことを考える。
もし $\{e\}^B(x_{2e}) \not\simeq 0$ なら $x_{2e} \notin A$ でよい。
$\{e\}^B(x_{2e}) \simeq 0$ であるとしよう。このとき (2) より、十分大きい s について

$$\{e\}_s^{B_s}(x_{2e}) \simeq 0 \tag{5}$$

となるはずである。そこで (5) のとき $x_{2e} \in A_{s+1}(\subseteq A)$ と定める。これにより (4) を満たすためには、$\{e\}_s^{B_s}(x_{2e})$ の計算が安定であればよい。つまり時刻 s 以降の t でもずっと $\{e\}_t^{B_t}(x_{2e}) \simeq 0$ であればよいが、それには

$$B|s = B_s|s \tag{6}$$

となっていれば十分である。言い換えると、一旦時刻 s で (5) であるがゆえに $x_{2e} \in A_{s+1}$ としたら、将来において B に s より小さい数を入れたくない。

ところが A と B の役割は対称なので、時刻 $t > s$ で $\{f\}_t^{A_t}(x_{2f+1}) \simeq 0$ となったら[2]、$x_{2f+1} \in B_{t+1}$ としたくなる。(6) を充たすためには $x_{2f+1} \geq s$ となっていてほしい。ところが計算が進行する前から予め $x_{2f+1} \geq s$ を充たすように大きく取っておくことはできない。そこで x_{2e} と x_{2e+1} 自体も時刻の進行とともに変動させていく。つまり（計算可能な）数列 $\{x_e^s\}_s$ をつくり

$$x_e = \lim_{s \to \infty} x_e^s$$

[2]時刻 t でこうなるのは、計算が進んで $\{f\}_t^{A_t}(x_{2f+1})$ の計算が終了したか、あるいは A_t にその計算に必要な情報が t までに盛られたかである

と定める。これは

$$\exists s[\forall t \geq s(x_e^t = x_e^s)\,\&\,x_e = x_e^s]$$

ということである。

状況を分かりやすくするため、異なる条件を満たす証拠は異なるようにしておく ($e \neq f \Rightarrow x_e^s \neq x_f^t$)：つまり

$$x_e^s \in \{\langle y, e\rangle : y \in \mathbb{N}\}$$

と定める。

後で定義する数列 $\{x_e^s\}_s$ は非減少列 $s < t \Rightarrow x_e^s \leq x_e^t$ となっていて、

$$(x_e^s)_0 \leq s \tag{7}$$

つまり $x_e^s = \langle y, e\rangle \Rightarrow y \leq s$ となっている。

すると上で述べた状況は、時刻 s で (5) であるがゆえに $x_{2e}^s \in A_{s+1}$ として同時にこれ以降、つまり $t > s$ で $x_{2f+1}^t > (x_{2f+1}^t)_0 \geq s$ となるように $\{x_{2f+1}^t\}_t$ を定義する。

しかしこれだけでは数列 $\{x_e^s\}_s$ は上で述べた意味での極限を持たないかもしれない。それはある $t > s$ で $x_{2f+1}^t \geq s$ が $\{f\}_t^{A_t}(x_{2f+1}^t) \simeq 0$ となり、この計算を安定にするために $u > t$ で $(x_{2e}^u)_0 \geq t > s \geq (x_{2e}^s)_0$ としてしまうと、数列 $\{x_{2e}^s\}_s$ が発散してしまう怖れがあるからである。

そこで、条件 R_e において、より小さい添字（＝プログラム）e に高い<u>優先度</u>(priority) をつける。そして R_{2e} について時刻 s で (5) となったら $x_{2e}^s \in A_{s+1}$ とすることで時刻 $s+1$ では R_{2e} が充たされるようにするが、(R_{2e+1} も同様)、より高い優先度の R_f ($f < 2e$) がそれを <u>覆す</u>(injure) つまり例えば $t > s$ で

$$2f + 1 < 2e\,\&\,\{f\}_t^{A_t}(x_{2f+1}^t) \simeq 0$$

となったら $x_{2f+1}^t < s$ であって (6) を壊すのだが、$x_{2f+1}^t \in B_{t+1}$ として、この時点での条件 R_{2e} は優先度の高い R_{2f+1} により覆されてしまい、$x_{2e}^{t+1} \geq t$ と新たに設定し直される。しかしこの「賽の河原積み」は各 e について有限回しかなされず、極限 $x_e = \lim_{s\to\infty} x_e^s$ が存在することが分かる（補題 4.2 参照)。

以上で証明の概略が説明し終わったので、正式な構成をしよう。以下で、有限集合 A_s, B_s と数 x_e^s, r_e^s を同時に s に関する帰納法で定義する。$s \mapsto (A_s, B_s, x_e^s, r_e^s)$ は計算可能となる。

r_e^s は $r_e^s = 0, 1$ で、$r_e^s = 1$ ということは「条件 R_e を充たすための変更が時刻 s かそれ以前になされ、かつ s に至るも尚その変更が覆されていない」ということを示すことになる。

初めに

$$A_0 = B_0 = \emptyset,\ x_e^0 = \langle 0, e\rangle,\ r_e^0 = 0.$$

条件 R_{2e} が時刻 $s+1$ で 注意喚起(require attention) とは

$$\{e\}_s^{B_s}(x_{2e}^s) \simeq 0 \,\&\, r_{2e}^s = 0 \tag{8}$$

となっていること。

これが B_s, $2e$ の代わりに A_s, $2e+1$ で成立するとき、R_{2e+1} が時刻 $s+1$ で注意喚起と呼ぶ。

時刻 $s+1$ での定め方：

1. $i \leq s$ なるどの i についても R_i が $s+1$ で注意喚起でないとき：何もしない、つまりどんな e についても

$$A_{s+1} = A_s, B_{s+1} = B_s,\ x_e^{s+1} = x_e^s,\ r_e^{s+1} = r_e^s.$$

2. ある $i \leq s$ について R_i が $s+1$ で注意喚起であるとき：そのような最小の i を取る。$i = 2e$ とする（$i = 2e+1$ なら以下で、A と B の役割を交換する)。

   (a)

$$A_{s+1} := A_s \cup \{x_{2e}^s\}, B_{s+1} = B_s,\ x_{2e}^{s+1} = x_{2e}^s,\ r_{2e}^{s+1} := 1.$$

   (b) R_{2e} より高い優先度の $R_j\,(j < 2e)$ については変更しない：

$$x_j^{s+1} = x_j^s,\ r_j^{s+1} = r_j^s.$$

   (c) R_{2e} より低い優先度の $R_j\,(j > 2e)$ については設定し直す：

$$x_j^{s+1} := \langle s+1, j\rangle > s,\ r_j^{s+1} := 0.$$

構成から分かる通り (7) が充たされ、よって

$$y \in A_s \cup B_s \Rightarrow \max\{(y)_0, (y)_1\} < s \tag{9}$$

まとめ

1. x_e^s が $t > s$ で変化するのは、ある $R_i\,(i < e)$ が t で注意喚起される場合に限る。

2. $r_e^s = 1$ が $t > s$ で $r_e^t = 0$ となるのは、ある $R_i\,(i < e)$ が t で注意喚起される場合に限る。

3. $x_{2e} = \langle y, 2e\rangle \in A$ なのは、ある $s > y$ で R_{2e} が注意喚起される場合に限る。

4. R_i が s で注意喚起になる最小の $i \leq s$ で $2e > i$ ならば、$x_{2e}^s = \langle s, 2e\rangle$, $r_{2e}^s = 0$.

まとめ

1. x_e^s が $t > s$ で変化するのは、ある $R_i\,(i < e)$ が t で注意喚起される場合に限る。

2. $r_e^s = 1$ が $t > s$ で $r_e^t = 0$ となるのは、ある $R_i\,(i < e)$ が t で注意喚起される場合に限る。

3. $x_{2e} = \langle y, 2e\rangle \in A$ なのは、ある $s > y$ で R_{2e} が注意喚起される場合に限る。

4. R_i が s で注意喚起になる最小の $i \leq s$ で $2e > i$ ならば、$x_{2e}^s = \langle s, 2e\rangle$, $r_{2e}^s = 0$.

補題 4.2 どの条件 R_i も高々有限個の時刻でしか注意喚起されない。

従って極限 $x_i = \lim_{s\to\infty} x_i^s$ が定義され、x_i が $A = \bigcup_s A_s, B = \bigcup_s B_s$ に関して条件 R_i を満たす。

証明. i に関する帰納法で注意喚起の有限性を示す。各 $j < i$ について R_j はある時刻以降では注意喚起されないから、十分大きい s を取ると、その時刻以降ではどの $R_j\,(j < i)$ も注意喚起されないとしてよい。s をこれを満たす最小の数とする。先ず $t > s \Rightarrow x_i^t = x_i^s = x_i$ である。

初めに R_i が $t > s$ で注意喚起されるとしたら $u > t \Rightarrow r_i^u = r_i^t = 1$ であるから、そのような機会は高々一回である。よって R_i は有限回しか注意喚起されない。

次に s の最小性より、$s > 0$ なら時刻 s においてある $R_j\,(j < i)$ が注意喚起されていたはずなので $x_i = x_i^s = \langle s, i\rangle$ となり、(9) より

$$x_i \notin A_s \cup B_s \,\&\, r_i^s = 0 \tag{10}$$

以下、$i = 2e$ として（$i = 2e + 1$ も同様）、x_{2e} が R_{2e} を満たすことを見る。

初めに R_{2e} がある $t + 1 > s$ で注意喚起である場合を考える。このとき $x_{2e} \in A_{t+1} \subseteq A$ で $\{e\}_t^{B_t}(x_{2e}) \simeq 0$ だが、$j > 2e, u > t$ について $x_j^u > t$ となっているので (6) $B|t = B_t|t$ であり、よって (2) より $\{e\}^B(x_{2e}) \simeq 0$ となり、条件 R_{2e} が充たされる。

次に R_{2e} はどんな $t+1 > s$ でも注意喚起されない場合を考える。すると (10) より $x_{2e} \notin A$ となり、また $r_{2e}^t = 0$ より $\{e\}_t^{B_t}(x_{2e}) \not\simeq 0$ なので（$\simeq 0$ なら R_{2e} が $t+1$ で注意喚起になる）(2) より $\{e\}^B(x_{2e}) \not\simeq 0$ となり、この場合も条件 R_{2e} が充たされる。 □

参考文献

[1] J. R. Shoenfield, Recursion Theory, Lecture Notes in Logic vol. 1(1993), Springer.

[2] R. I. Soare, Recursively Enumerable Sets and Degrees, Perspective in Mathematical logic(1987), Springer.