Title	経済数学覚書5 : 計算論, 不動点定理および囚人のジレンマ
Sub Title	Notes on math of economics 5 : recursion theory, fixed point theorem and prisoner's dilemma
Author	中山, 幹夫(Nakayama, Mikio)
Publisher	慶應義塾経済学会
Publication year	2025
Jtitle	三田学会雑誌 (Mita journal of economics). Vol.117, No.3 (2025. 2) ,p.347 (81)- 359 (93)
JaLC DOI	10.14991/001.20250201-0081
Abstract	
Notes	解説
Genre	Journal Article
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=AN00234610-20250201-0081

「三田学会雑誌」117 巻 3 号（2025 年 2 月）

解　説

経済数学覚書 5
――計算論，不動点定理および囚人のジレンマ――

中山幹夫 *

1.　はじめに

経済学における数学的方法としては，連続量を扱う解析学が主として使われてきた。ゲーム理論もまたその流れに沿って発展してきたが，初期においてはたとえば Rabin [14] や Jones [6] などの論理学者によるいわゆる算術ゲームにおける必勝戦略が「計算不可能」であるという研究がある。これらの研究では連続的変数を扱う関数ではなく，自然数から自然数への関数のみを対象とする古典的**帰納理論**（**Recursion Theory**）あるいは**計算論**（**computability**）と呼ばれる数学基礎論的方法が使われている。

ゲーム理論においてはその後も「限定合理性」に関する研究の展開の中で，計算論による限定合理的プレイヤーの行動についての研究も行われてきた。たとえば Howard [5], Anderlini [1], Prasad [13], Canning [3], Knoblauch [7]，Nachbar and Zame [8] など，また一般均衡に関するものとしては Richter and Wong [15] などがあるが，Howard [5] を除いて，これらの Rabin [14] や Jones [6] をはじめとするほとんどの研究は，最適戦略や Nash 均衡の計算不可能性，あるいは Nash 均衡の存在の決定不能性（e.g. Prasad [13]），また一般均衡の計算不可能性を主張する内容となっている。

Howard [5] は，しかし，これらの研究とは逆に可能性を示唆するプレイヤーのモデルを提示するという，計算論の応用としてはまれな研究である。すなわち，囚人のジレンマにおいて「相手が自分と同一であることを認識する能力＝**自己認識能力**」を備えた戦略を提唱し，この能力をもつ戦略

査読者からの建設的なコメントにより記述を改善できたことに感謝いたします。

＊　慶應義塾大学名誉教授

同士では協力行動が実現するという可能性について論じた。実際，コンピュータ言語の BASIC によるプログラムを作成し，これが自己認識を実行して協力行動が実現することを示している。

本稿では，この自己認識という**自己言及**（**self-reference**）を本質とする行動をまず定式化するという目的で，初歩的計算論の準備のもとに**計算論の不動点定理**に着目する。さらにこれを「**血縁認識**」に拡張した Nakayama [9] の議論を再構成して，できるだけ self-contained に再現することを試みる。計算論は**チューリング・マシン**による計算を前提としているが，ここでは Cutland [4] に従ってそれと同等でより扱いやすい **URM**（**Unlimited Register Machine**）を基礎に置き，証明などにはこの URM プログラムによる計算も取り入れて，本稿の前編「経済数学覚書 4」の末尾で触れたように計算論の初等的な経済学的応用例として述べてみたい。

2. 計算可能関数

まず，次の例をみてみよう。これは Cutland [4] の 8 ページに述べられている例であるが，
円周率 π の 10 進展開において，$n \in \mathcal{N} := \{0, 1, 2, 3, \ldots, n, \ldots\}$ とするとき，

$$g(n) := \begin{cases} 1 & \text{ちょうど } n \text{ 個の連続する 7 がある場合} \\ 0 & \text{そうでない場合} \end{cases}$$

と定義される関数 g を考える。多くの数学者は g を well-defined とみなすように思われるが，もし，ある $n > 1$ についてちょうど n 個の連続する 7 が存在しない場合，値 0 を得ることができるだろうか。そのためには π の近似式によって存在しないことを確かめなくてはならないが，これを有限のステップで完了することは不可能である。つまり，値 0 は有限のステップでは得られないので，g は計算できるとは言えないだろう。本稿ではすでに述べたように有限の手順として URM（Unlimited Register Machine）と呼ばれるチューリング・マシンと同等な理論上の「計算機」を想定しており，URM で計算できない関数は計算可能ではない。

2.1. URM（Unlimited Register Machine）

この 計算機にはまず無限個のレジスター R_1, R_2, R_3, ..., R_n, ... が用意されていて，各レジスター R_i には自然数 r_i が格納される。さらに 4 種類の命令 ***Zero, Successor, Transfer, Jump*** が備わっており，この計算機のプログラム P はこれらの命令を有限個組み合わせたリストである。各々の命令は次のような動作を実行する：

- ***Zero*** は $n = 1, 2, 3, \ldots$ について，命令 $Z(n)$ により R_n の数値を 0 に更新する。

- ***Successor*** は $n = 1, 2, 3, \ldots$ について，命令 $S(n)$ によって R_n の数値に 1 を加える。
- ***Transfer*** は $m, n = 1, 2, 3, \ldots$ に対し，命令 $T(m, n)$ によって R_n の数値を R_m の数値と同じ値 r_m に更新し，R_m の数値は r_m のまま残す。
- ***Jump*** は $m, n, q = 1, 2, 3, \ldots$ に対し，命令 $J(m, n, q)$ によって，もし $r_m = r_n$ ならばプログラム P の q 番目の命令にジャンプし，もし $r_m \neq r_n$ ならば P における次の命令を実行する。

たとえば演算 $f(x, y) = x + y$ は，各レジスターに初期配列として $\boldsymbol{r_1 = x,\ r_2 = y,\ r_3 = 0,\ r_4 = 0,\ r_5 = 0,\ \cdots}$ と与えておけば，次のプログラム $\boldsymbol{P := I_1, I_2, I_3, I_4}$ で実行できる：

$$\boldsymbol{I_1 = J(3, 2, 5),\ \ I_2 = S(1),\ \ I_3 = S(3),\ \ I_4 = J(1, 1, 1).}$$

まず，初期配列に対して I_1 では $y = 0$ ならば I_5 は存在しないので計算は停止し，結果はレジスター R_1 に格納されている値 x となる。$0 \neq y$ ならば I_2 へ移り $r_1 = x + 1$, I_3 で $r_3 = 1$ として I_1 へ戻る。この計算過程において状態が $r_1 = x + k$, $r_2 = y$, $r_3 = k$, $r_4 = 0$, $r_5 = 0$, $\ldots$ であるとすると，I_1 で $k = y$ ならば $I_5 :=$ 停止，$k \neq y$ ならば I_2 へ進む。I_2 では $r_1 = (x + k) + 1$ と更新し，I_3 では $r_3 = k + 1$ として I_4 により I_1 へ戻る。これを繰り返して $k = y$ となった時点で計算は停止し，R_1 に格納されている値は $x + y$ となってこれが計算結果となる。(1)

以下では，ある URM プログラムで関数 $f^{(k)} : \mathcal{N}^k \to \mathcal{N}$ を計算できるとき，すなわち，任意の $a := (a_1, a_2, \ldots, a_k)$ について a が $f^{(k)}$ の定義域の中にあり，初期配列 $a_1, a_2, \ldots, a_k, 0, 0, \ldots$ に対して最終的にレジスター R_1 に値 $r_1 = b \in \mathcal{N}$ が格納されたならば，$f(a) = b$ となって k 変数関数 $f^{(k)}$ は計算可能であると言う。$f^{(k)}$ の定義域が $\mathcal{N}^k$ 全体であるならば $f^{(k)}$ は**全計算可能関数**であると言う。さらに，1 変数関数 $f^{(1)}$ については $k = 1$ を省略してたんに f と書くことにする。

2.2. ゲーデル数（Gödel number）と s-m-n 定理

すでに述べたように，URM プログラム P は 4 種類の命令で作る組み合わせの有限列である。$x + y$ の計算は，初期配列 $x, y, 0, 0, \ldots$ のもとで $\boldsymbol{J}$ と $\boldsymbol{S}$ の 2 種類によるプログラム $J(3, 2, 5)$, $S(1)$, $S(3)$, $J(1, 1, 1)$ で計算可能であった。

いま，すべての URM プログラムの集合を $\mathcal{P}$ とすると，自然数の集合 $\mathcal{N}$ との間に**両方向に計算可能な全単射**（**bijection**）γ を作ることができる。(2) この γ は，不完全性定理の証明のために必要な操作として初めて導入した K.Gödel の名をとって**ゲーデル数**（**Gödel number**）と呼ばれている。

---

(1) 最後の命令の実行後，次に実行すべき計算が無ければそこで計算は停止することになる。

(2) 詳細は Cutland op.cit. pp. 72–84. また，中山 [10] pp. 44–46 には簡略化した解説がある。

本稿ではこの γ を前提として，$\gamma(P) \in \mathcal{N}$ を URM プログラム P の**指標**ないし**番号**と呼ぶ。

すると，番号 $\gamma(P) \in \mathcal{N}$ を小さいものから順に並べた，すべての 1 変数計算可能関数の（重複を許した）一覧：

$$\phi_0,\ \phi_1,\ \phi_2,\ \ldots,\ \phi_n,\ \ldots \tag{1}$$

を得ることができる。また，k 変数計算可能関数 $\phi_n^{(k)}$ の一覧は番号 $n = \gamma(P)$ のプログラムに k 個の変数を入力すれば得られる。この一覧が重複を許していることは，たとえば関数 $g(x, y) = x$ の URM プログラムは，初期配列 $x, y, 0, 0, 0, \ldots$ のもとで，一個の命令 $Z(2)$ や $S(2)$，あるいは $J(1, 1, 2)$ によっても計算可能であることからわかる。

次に述べる定理は，計算可能関数 $f(x, y)$ を計算するプログラム番号を入力データの x から $k(x)$ として与える**全計算可能関数 $k(x)$** の存在を主張するもので，後の不動点定理の証明に使う。

s-m-n 定理. 関数 $f(x, y)$ が計算可能ならば

$$\boldsymbol{f(x, y) \simeq \phi_{k(x)}(y)}$$

をみたす全計算可能関数 $k(x)$ が存在する[(3)]。

証明. 任意に固定された自然数 a について，$k(a)$ を計算可能関数 $f(a, y)$ を計算する URM プログラム Q_a の番号であるとし，初期配列を

$$\boldsymbol{r_1 = y,\ \ r_2 = 0,\ \ r_3 = 0,\ \ r_4 = 0,\ \ \ldots}$$

としよう。また，f を計算する URM プログラムを $\boldsymbol{F}$ としてこれを末尾に付け加えて Q_a を

$$\boldsymbol{T(1, 2)\ \ Z(1)\ \ S(1)\ \ \cdots\ \ S(1)\ \ F}$$

と定義する。ここで，上の $S(1) \cdots S(1)$ は $S(1)$ を a 回繰り返すことを表す。すると，レジスターの状態は

$$\boldsymbol{r_1 = a,\ \ r_2 = y,\ \ r_3 = 0,\ \ r_4 = 0,\ \ \ldots}$$

となる。$k(a)$ はこの状態から関数 $f(a, y)$ を計算するプログラム Q_a の番号であり，ゲーデル数の全単射 γ は計算可能であるから関数 $k(a)$ は計算可能。こうして，任意の a について

---

(3) Cutland op.cit. p. 81, 4 The s-m-n Theorem. なお記号 $\simeq$ は両辺が定義されていれば $=$, そうでなければ両辺は undefined とするという意味である。

$$\phi_{k(a)}(y) \simeq f(a,y)$$

が得られる。 □

2.3. 不動点定理と自己言及

解析学で関数 f の不動点と言えば $f(x)=x$ をみたす変数 x のことであるが，計算論では不動点とは以下に述べるように $\phi_e=\phi_{f(e)}$ をみたす e であって，e と $f(e)$ は同じ関数を計算する URM プログラムの番号という意味である。そのため e は f の疑似不動点あるいは**プログラム不動点**と呼ばれることがある。

不動点定理． $f:\mathcal{N}\to\mathcal{N}$ を**全**計算可能関数とすると $\boldsymbol{\phi_{f(e)}=\phi_e}$ をみたす番号 e が存在する[4]。

証明． プログラム番号 $f(\phi_x(x))$ をもつ関数 $\phi_{f(\phi_x(x))}(y)$ は計算可能だから，$g(x,y):=\phi_{f(\phi_x(x))}(y)$ に s-m-n 定理を適用すると，全計算可能関数 $s(x)$ が存在して，

$$\phi_{f(\phi_x(x))}(y) = g(x,y) \simeq \phi_{s(x)}(y).$$

ここで，もし $\phi_x(x)$ が undefined ならば上式左辺は undefined であるとする。$s=\phi_m$ となる任意の番号 m をとり，上式を

$$\phi_{f(\phi_x(x))}(y) \simeq \phi_{\phi_m(x)}(y)$$

と書いて $x=m$，また ϕ_m は全計算可能だから $n=\phi_m(m)$ とすると

$$\phi_{f(n)}(y)=\phi_n(y)$$

が得られる。 □

系 1. 不動点は無限個存在する。

証明． まず任意の $m\in\mathcal{N}$ に対し，$\phi_j\neq\phi_n \quad (\forall n\le m)$ をみたす $j\in\mathcal{N}$ をとり，次のように全計算可能関数 $h:\mathcal{N}\to\mathcal{N}$ を定義する[5]：

---

(4) Cutland op.cit. p. 200 ではこれを第 2 帰納定理としているが，ここでは Odifreddi [12] p. 152 に従って不動点定理と呼んでおく。

(5) Bridges [2] p. 159. 5.14.17, または Cutland op.cit. p. 202, 1.3 Corollary.

$$h(i) = \begin{cases} j & \text{if} \quad i \leq m \\ f(i) & \text{if} \quad i > m \end{cases}$$

ただし f は不動点定理における全計算可能関数である。すると，不動点 i が存在して $\phi_i = \phi_{h(i)}$ であるが，ここでもし $i \leq m$ だとしたら関数 h の定義により $h(i) = j$ であり，それゆえ $\phi_i = \phi_j$ となって矛盾が生じる。ゆえに，$i > m$, $h(i) = f(i)$ であるから任意の m に対し，$i > m$ をみたす不動点 $\phi_i = \phi_{f(i)}$ が存在する。 □

第 2 帰納定理. $g : \mathcal{N}^2 \to \mathcal{N}$ を計算可能関数とすると，

$$\boldsymbol{\phi_e(y) \simeq g(e, y)}$$

をみたす番号 e が存在する[(6)]。

証明. 関数 $g(x, y)$ は計算可能だから，s-m-n 定理より $\phi_{s(x)}(y) \simeq g(x, y)$ をみたす全計算可能関数 s が存在する。ゆえに，不動点定理より s の不動点 e をとれば

$$\phi_e(y) = \phi_{s(e)}(y) \simeq g(e, y)$$

が得られる。 □

この定理は不動点定理と同値であることが知られているが，本稿ではこのように不動点定理の系として得られれば十分である。この定理の意味するところは，プログラム e を $\phi_e(y) \simeq g(e, y)$ によって定義できるということ，すなわち，自己言及的な定義が可能であるということである。たとえば，関数 $g(x, y) = x$ は計算可能であるから

$$\phi_e(y) = g(e, y) = e$$

をみたすプログラム e が存在する。「e とは e である」と読むと無意味な定義であるが，e は自分自身を複製するプログラムであり，自己複製ウイルス（self-replicating virus）とも呼ばれるプログラムである。

さらに，次のように定義される関数 $g : \mathcal{N}^2 \to \mathcal{N}$ を考えよう。

---

(6) ここでも Odifreddi op.cit. p. 156 に従ってこの命題を第 2 帰納定理と呼んでおこう。

$$g(x,y) := \begin{cases} 1 & \text{if} \quad x = y \\ 0 & \text{if} \quad x \neq y \end{cases}$$

この関数 g は，初期配列を $\boldsymbol{r_1 = x,\ r_2 = y,\ r_3 = 0,\ r_4 = 0,\ \ldots}$ として URM プログラム

$$\boldsymbol{J(1,2,4)\quad Z(1)\quad J(1,1,7)\quad J(1,1,5)\quad Z(1)\quad S(1)}$$

で計算可能である[(7)]。ゆえに第 2 帰納定理によって不動点 s が存在して

$$\phi_s(y) = g(s,y) = \begin{cases} 1 & \text{if} \quad s = y \\ 0 & \text{if} \quad s \neq y \end{cases} \tag{2}$$

となる。このプログラム s は，入力 y が自分 s と同じならば 1，そうでなければ 0 を出力することを示しており，1 を協力行動 c，0 を非協力行動 d とみなせば囚人のジレンマにおける Howard [5] の**自己認識プレイヤー**として定義することができる。

3. 血縁認識と囚人のジレンマ

Howard [5] の自己認識プレイヤーは，上の(2)で定義したように自分と相手のプログラム番号が一致しなければ協力しない。しかし，計算可能関数の一覧表(1)には重複があるので，同じ関数は異なるプログラムでも計算可能であるが，自己認識プレイヤーはそのような相手を認識せず非協力行動をすることになる。この意味で自己認識行動は非効率的であると言える。

本節では協力の相手をもっと広いプログラムの集合に拡張することを考える。目的はプログラムの番号の集合 M をとり，$x, y \in M$ ならば x と y は協力行動をするように定義することである。そのための準備として次の用語を定義しておこう。x と y の**関係**ないし x と y に関する**述語**（**predicate**）を一般に $R(x,y)$ で表し，特性関数

$$g(x,y) = \begin{cases} 1 & \text{if} \quad\quad R(x,y) \\ 0 & \text{if} \quad \neg\, R(x,y) \end{cases}$$

が計算可能であるとき，$R(x,y)$ は**決定可能**（**decidable**）あるいは**帰納的**（**recursive**）などと言う。集合 $S \subsetneq \mathcal{N}$ については，メンバーシップが決定可能であるとき，つまり，述語 $x \in S$ が決定可能ならば，S は**帰納的集合**（**recursive set**）であると言う。

---

(7)　$S(1)$ の実行後，次に実行する命令がないので計算は停止することに注意。

3.1. Rice の定理と血縁プログラム

さて，$e \in \mathcal{N}$ を任意のプログラムの番号とし，集合

$$I_e := \{x \in \mathcal{N} \mid \phi_x = \phi_e\} \tag{3}$$

を目的の集合と定義したい。しかし，I_e は関数 ϕ_e と同じ値をとる関数の番号すべての集合であるが，まず，述語「$\phi_x = \phi_e$」は帰納的ではない。関数の同等を有限のステップで確かめることはできないからである。次の定理はこのような事実を一般的に主張する定理である。

Rice の定理. 1 変数計算可能関数の全体 $\mathcal{I}_1$ の空でない真部分集合を $\mathcal{A}$ とし，$A := \{x \mid \phi_x \in \mathcal{A}\}$ とすると集合 A は帰納的でない[(8)]。

証明. $a \in A$ および $b \notin A$ とする。A は帰納的であるとすると，関数

$$f(x) = \begin{cases} a & \text{if} \quad x \notin A \\ b & \text{if} \quad x \in A \end{cases}$$

は全計算可能。すると，すべての x に対し，$x \in A \iff f(x) \notin A$. しかし，不動点定理によってある n が存在して $\phi_n = \phi_{f(n)}$ であるから A の定義により $n \in A \iff f(n) \in A$. これは矛盾である。 □

集合 $\{\phi_x \mid \phi_x = \phi_e\}$ は $\mathcal{I}_1$ の空でない真部分集合であるから，$\{\phi_x \mid \phi_x = \phi_e\} = \mathcal{A}$ とすると Rice の定理により集合 $I_e = \{x \mid \phi_x \in \mathcal{A}\ \} = A$ は帰納的ではない。それゆえ，$x \in I_e$ であるかどうか，つまり，x が I_e のメンバーであるかどうかは決定可能ではない。そこで，帰納的な集合を得るため I_e の空でない真部分集合をとり，次のように定義しよう。

$$I_e^* := \{e, e_1, e_2, e_3, \ldots\} \subsetneq \{y \mid \phi_y = \phi_e\} \quad \text{ただし} \ \ e < e_1 < e_2 < e_3 < \cdots \tag{4}$$

仮定 1. I_e^* は**帰納的に枚挙可能**（**recursively enumerated**）である。

I_e^* が帰納的に枚挙可能とは, ある**単調増加全計算可能関数** f によって $f(0) = e$, $f(1) = e_1$, $f(2) = e_2$, $f(3) = e_3$, ... となること，つまり，I_e^* は f の値域になっているということである。この仮定

(8) Cutland op.cit.Theorem 1.6 p. 203.

により I_e^* のメンバーシップは次に示すように決定可能となる：

補題 1. I_e^* は帰納的集合である。

証明. f は単調増加であるから $n \leq f(n) \quad \forall n \in \mathcal{N}$。ゆえに，$y := f(n)$ および f の値域を $E(f)$ として，

$$y \in I_e^* \iff y \in E(f) \iff \exists\, n \leq y\ (f(n) = y).$$

右辺の述語は，与えられた y に対して y 以下のすべての自然数 n について，決定可能な述語「$f(n) = y$」を成り立たせる n を探すことを述べており，有限のステップで決定可能である（**bounded search**）。ゆえに左辺も決定可能である。 □

こうして任意のプログラム x が I_e のメンバーであるか否かは確定することになる。なお，仮定 1 において I_e^* の番号が単調増加的としていることは，番号 e のプログラムにその出力に影響しない余分な命令を順次 1 個，2 個，3 個，... と追加していけば，同じ関数を計算するプログラムを任意有限個生成できることを主張する Padding Lemma と[(9)]，それによりプログラム番号の増加列が生じることと矛盾しない。

さて，集合 I_e^* に次の仮定を追加しよう：

仮定 2. $\forall x, y \in \mathcal{N}$ に対して，$I_x^* \cap I_y^* \neq \emptyset \Longrightarrow I_x^* \subseteq I_y^*$ or $I_y^* \subseteq I_x^*$.

この仮定により $x \in I_e^* \iff I_x^* \subseteq I_e^*$ であることがわかる。$x \in I_e^*$ かつ $I_e^* \subsetneq I_x^*$ だったら $e \leq x$ かつ $x < e$ となり矛盾するから $x \in I_e^*$ ならば $I_x^* \subseteq I_e^*$ でなければならないからである。こうして集合 I_e^* のメンバーは e の**子孫**と呼ぶことができるだろう。すなわち「x が e の子孫ならば x の子孫は e の子孫」であり，$e_j,\ e_k \in I_e^*$ ならば e_j と e_k は互いに**血縁プログラム**であると言うことができる。

3.2. 帰納的血縁関係と血縁認識プレイヤー

プログラム x, y について，述語 $K^*(x, y)$ を次のように定義する：

$$K^*(x, y) \iff \exists\, w \in \mathcal{N} \ \text{ s.t. } \ x \in I_w^* \ \wedge \ y \in I_w^* \tag{5}$$

I_e^* の作り方(4)から $w \leq x,\ y$ であるから，右辺は「x と y は**先祖** w を共有する子孫である」と読むことができるので，これにより $K^*(x, y)$ を「x と y は血縁である」ことを述べる述語，つまり**血**

(9) Odifreddi op.cit. p. 131, Proposition II.1.6.

縁関係と呼ぼう。すると，

命題 1. 血縁関係 $K^*(x,y)$ は**決定可能な同値関係**である。

証明. まず，同値関係であることは，反射性：$K^*(x,x)$ と対称性：$K^*(x,y) \Rightarrow K^*(y,x)$ は明らかだから，推移性を示せばよい。$K^*(x,y)$, $K^*(y,z)$ より $\exists\, w,v \in \mathcal{N}$ s.t. $x,y \in I^*_w$ and $y,z \in I^*_v$. すると $y \in I^*_w \cap I^*_v$ だから仮定 2 より $I^*_w \subseteq I^*_v$ or $I^*_v \subseteq I^*_w$. ゆえに $x,\ z \in I^*_v$ or $x,\ z \in I^*_w$ であるから $K^*(x,z)$ が成り立つ。

決定可能性については，補題 1 より $R(x,y\,;w) := x \in I^*_w \wedge y \in I^*_w$ は決定可能な述語である。また，I^*_e の作り方(4)より $w \leq x,\ y$ だから，$\mu := \min\{x,\ y\}$ とすると

$$\boldsymbol{K^*(x,y) \iff \exists\, w \leq \mu\ R(x,y\,;w)}$$

の右辺の真偽は**有限個**の $e \leq \mu$ について決定可能な述語 $R(x,y\,;e)$ の真偽をチェックすることで確定できるから決定可能である（**bounded search**）。ゆえに $K^*(x,y)$ は決定可能（帰納的）である。

□

ここで，囚人のジレンマの**血縁認識プレイヤー**を次のように定義する。

定義 1. 囚人のジレンマにおいて，プログラム $k \in \mathcal{N}$ が任意のプログラム $y \in \mathcal{N}$ に対して

$$\phi_k(y) = \begin{cases} 1 & \text{if} \quad K^*(k,y) \\ 0 & \text{if} \quad \neg\, K^*(k,y) \end{cases}$$

をみたすとき，k を**血縁認識プレイヤー**（**Kin-Recognition Player, KRP**）と呼ぶ。ただし，1 を協力行動 c，0 を非協力行動 d と解釈する。

このように，プログラム k が KRP ならば，血縁プログラムとは協力し，そうでない相手とは協力しない。また，同値関係より協力する血縁プログラムは KRP であり，$\phi_k = \phi_w$ だから先祖 w も当然 KRP である。

Theorem 1. 無限個の血縁認識プレイヤーが存在する。

証明. 血縁関係 $K^*(x,y)$ は命題 1 より決定可能だから特性関数

$$g(x,y) := \begin{cases} 1 & \text{if} \quad K^*(x,y) \\ 0 & \text{if} \quad \neg\, K^*(x,y) \end{cases}$$

は計算可能である。ゆえに，第 2 帰納定理より不動点 k が存在して

$$\phi_k(y) = g(k,y) = \begin{cases} 1 & \text{if} \quad K^*(k,y) \\ 0 & \text{if} \quad \neg\, K^*(k,y) \end{cases}$$

となるから，k が求める KRP である。

また，系 1 より不動点の個数は無限個となる。 □

こうして，one-shot の囚人のジレンマにおいて血縁認識プレイヤーが存在し，血縁認識プレイヤー同士の協力が実現することが示された。なお，I_e^* の定義式(4)において $I_e^* = \{e\}$ とすると $K^*(x,y)$ の定義式(5)は

$$K^*(x,y) \iff x = y$$

となって，血縁認識は自己認識に「退化」する。系 1 より自己認識プレイヤーも可算無限個存在するので，血縁認識プレイヤーの存在が無限個であることはあまり意味がなく，自分と同じでなくてもより広い範囲の相手との協力の可能性が明らかになったことがこの定理の意義であると言える。

最後に，定理 1 の系として，次のような多少とも奇妙な行動をするプログラムの存在について触れておこう。

系 2. プログラム k は KRP であるとすると，$\neg K^*(k,z)$ であって

$$\phi_z(y) = \begin{cases} 1 & \text{if} \quad K^*(k,y) \\ 0 & \text{if} \quad \neg\, K^*(k,y) \end{cases} \tag{6}$$

をみたすプログラム z が存在する。

証明. まず，集合 $\{y \mid K^*(k,y)\} \subsetneq I_k$ において $\{y \mid K^*(k,y)\}$ は帰納的であるが I_k は定義(3)と Rice の定理より帰納的ではないから $\emptyset \neq I_k \setminus \{y \mid K^*(k,y)\}$ である。すると

$$z \in I_k \setminus \{y \mid K^*(k,y)\}, \quad \neg K^*(k,z) \quad \text{かつ} \quad \phi_z = \phi_k$$

をみたす z が存在する。ゆえに，$\phi_z(y) = \phi_k(y)$ であるから(6)式が成り立つ。 □

このプログラム z は，相手 y が自分とは血縁でない k の血縁であるときのみ協力し，しかも $\phi_z(z) = \phi_k(z) = d$ となるので，相手が自分と同じであっても協力しない。この意味で「k 従属的利他主義者」とでも言うべきプログラムであり，KRP に付随して必然的に存在している。しかし，$\phi_z(k) = c$ かつ $\phi_k(z) = d$ であるから (z, z) はナッシュ均衡にはならず，この奇妙なプログラム z は進化ゲーム論的には KRP に駆逐される運命にあると言えるだろう。

4. おわりに

計算論が経済数学において話題にされることは現状ではあまり無いものと思われる。解析学と異なり，離散的変数が主要な対象であることに加えて，冒頭でも述べたように不可能性を導く研究が多いことがその理由かもしれない。同じ不可能性でも Arrow の不可能性定理はきわめて有名であるが，Richter and Wong [15] による一般均衡の計算不可能性はそれほどでもないのも，使用されている**計算可能実数**の理論が経済学者にはなじみのない理論であることが主な理由ではないだろうか。

それでも，Velupillai [17] による *Computable Economics* や *Computability, Complexity and Constructivity in Economic Analysis* などの経済分析への応用を目指す著書や編著もあり，また最近の AI の発展との関連において計算論とかかわりの深い machine learning 理論などへの関心が高まっていくことが期待される。

さらに，最近の経済学や OR，ゲーム理論などにおける離散的変数の取り扱いについて言えば，離散凸解析やマーケットデザインなどでは離散的変数も主な分析対象であり，とくにマッチング理論はノーベル経済学賞を獲得したこともあって，経済数学としても伝統的解析学と並んで離散的変数を分析に取り込むことが今後ますます必要になっていくものと思われる。

参 考 文 献

[1] Anderlini, L., “Some notes on Church’s thesis and the theory of games,” *Theory and Decision*, **29**, (1990), 19–52.

[2] Bridges, S. D., *Computability – A Mathematical Sketchbook* –, Springer-Verlag, New York, (1994).

[3] Canning, D., “Rationality, computability, and Nash equilibrium,” *Econometrica*, **60** (1992), 877–888.

[4] Cutland, N.J., *Computability*, Cambridge University Press, Cambridge, (1980).

[5] Howard, J. V., “Cooperation in the Prisoner’s Dilemma,” *Theory and Decision*, **24** (1988), 203–213.

[6] Jones, J. P., “Some undecidable determined games,” *International Journal of Game Theory*, **11** (1982), 63–70.

[7] Knoblauch, V., “Computable strategies for repeated Prisoners’ Dilemma,” *Games and Economic Behavior*, **7** (1994), 381–389.

[8] Nachbar, J. H. and W.R.Zame, “Non-computable strategies and discounted repeated games,” *Economic Theory*, **8** (1996), 102–122.
[9] Nakayama, M., “Mutual cooperation and unilateral altruism in a one-shot Prisoner’s Dilemma – a computability approach –,” *KUMQRP Discussion Paper Series*, **DP2005-003** (2005).
[10] 中山幹夫「ゲームと戦略の計算可能性について」『三田学会雑誌』91 巻，4 号，1999 年。
[11] 中山幹夫「経済数学覚書 4——帰納法雑録——」『三田学会雑誌』115 巻，3 号，2022 年。
[12] Odifreddi, P., *Classical Recursion Theory,* North-Holland, Amsterdam, (1992).
[13] Prasad, K., “Computability and randomness of Nash equilibrium in infinite games,” *Journal of Mathematical Economics*, **20**, (1991), 429–442.
[14] Rabin, M. O., “Effective computability of winning strategies,” In M.Dresher et al. eds. *Contributions to the Theory of Games III, Annals of Math. Studies,* Princeton University Press, Princeton, (1957).
[15] Richter, M. K. and Kam-Chau Wong, “Non-computability of competitive equilibrium,” *Economic Theory*, **14** (1999), 1–27.
[16] Velupillai, K. V., *Computable Economics,* Oxford University Press, New York, (2000).
[17] Velupillai, K. V., *Computability, Complexity and Constructivity in Economic Analysis,* Blackwell Publishing, Oxford, (2005).