人　工　知　能　102－6
（1995．11．7）

# 利己的なエージェントによる協調的性向の学習

伊藤 昭

郵政省通信総合研究所

自らの判断で行動し，自己の行動に責任を持つ自律エージェントは，他の自律エージェントとどのようにつき合っていったら良いのだろうか．協調してある行動をとった方が有利な状況下では，自己の利益を犠牲にしてでも協調することが可能であろうか．何の条件も付けなければ，答えは否定的であろう．Dawkinsの答えは，遺伝子が共有されていればYes，というものであった．Axelrodの答えは，繰り返し同じ相手と対戦するのであればYes，であった．我々は第3の条件，「他のエージェントの過去の行動がわかるのなら」を取り上げ，囚人のジレンマゲームをモデルに，利己的なエージェントが如何に行動すべきかを調べてみた．その結果，エージェントは適切な環境下では，他エージェントとのつき合いの中で，協調することを学習できることがわかった，

# How Selfish Agents Learn to Cooperate?

Akira Ito

Communications Research Laboratory

Iwaoka, Nishiku, Kobe, 651-24, Japan

How should an autonomous agent, which acts at its own will and takes the responsibility for its own actions, interact with other agents? Can it cooperate against its individual gains, if cooperation is favorable or profitable for the society as a whole? The Dawkins' answer is "yes, so long as the genes are shared among them". The Axelrod's answer is "yes, so long as the interaction occurs repetitively between same agents". The third condition: "if information on the past behaviors of other agents is known" is taken up, and how selfish agents should behave under such a situation is investigated. The Prisoner's Dilemma Game is employed as a model of interaction. It is found that agents can learn to cooperate under a proper circumstances.

## 1 はじめに

自らの判断で行動し，自己の行動に責任を持つ自律エージェントは，他の自律エージェントとどのようにつき合っていったら良いのだろうか．もちろん，「自律エージェントとは何か」については，必ずしも意見の一致があるわけではないので [Ishida 95]，まず我々の立場を述べることから始める．

**定義** 自律エージェントとは，外部の観測者から見てその内部機構がアルゴリズミックに理解できない自律システムのことである．また，エージェントの目的，意図，価値観，といった概念は，観測者が内部機構の理解できないエージェントをモデル化するために導入したヒューリスティックである．

自律エージェント同士のつき合いにおいては，相手がどのような行動規則に従っているかを，予め知ることはできない．自律エージェントにとって唯一可能なことは，相手の過去の行動から相手のモデルを推測することである．このとき一般には，

・共通の価値観の存在を仮定してはいけない，

・相手の合理的行動を仮定してはいけない．

しかしながら，相手をモデル化するに当たって，相手が「合理的エージェント」であるという仮定は，しばしば採用されてきた．相手の合理性を仮定できるためには，

・相手はある価値観に従って行動していること，

・相手にはアルゴリズム計算能力があること，

の二つの条件が必要となる．

このような仮定は，その背後にに「合理的な設計者」を予想すれば，ある程度許されるかも知れない．しかしながら，過度の合理性付与はしばしば惨めに破綻する．また，相手がどのような価値観に従って行動しているかを直接調べる方法はない．とはいえ，もし相手の価値観が全く何もわからなければ，行動の予測は殆んど不可能である．そこで，我々は共通の価値観が可能となる一つの世界を導入する．それは，生態系のように，物理的法則に従って個体数が増減する世界である．

**定義** 自律エージェントの共通の価値とは，自己の子孫をできるだけ増加させることである．

もちろん，これだけでは単なる定義であり，「共通の価値」に特別の意義はない．重要なことは次が成り立つことである．

**自然淘汰** 子孫の増加という共通の価値を追求しない種は，長期的にはその個体数を減少させ，いずれは絶滅してしまう．逆に言えば，長期的に生き残っている個体は，子孫の増加という共通の価値を追求しているはずである．

このような世界において，エージェントが「自らの判断で行動する」ということは，単にそれが内部の規則に基づいて動作するということである．また，「自らの行動に責任をとる」ということは，自然淘汰という物理的法則の存在を述べているにすぎない．このような自律エージェントは，まさに Dawkins が「利己的」[Dawkins 76] と呼んだものであり，それはまた近代経済学が土台とてきした自由な経済人（ホモ・エコノミカス）とも同じ概念である．

では，ここで定義された意味で利己的なエージェントは，長期的な視野で協調して，社会を作ることができるのだろうか．そのために必要な条件があるとすれば，それはどのようなものであろうか．もう少し具体的に言えば，協調してある行動をとった方が有利な状況下では，自己の利益を犠牲にしてでも協調することが可能であろうか．何の条件も付けなければ，答えは否定的であろう．Dawkins の答えは，遺伝子が共有されていれば Yes，というものであった．Axelrod[Axelrod 84] の答えは，繰り返し同じ相手と対戦するのであれば Yes，であった．しかしながら，人間社会を見てみると，この両者が当てはまらない状況でも協調が行われているようにも見える．そこで我々は第 3 の条件，「他のエージェントの過去の行動がわかるのなら」を取り上げ，これに Yes と答えたいのである．

## 2 利己的なエージェントのつき合い方

我々がここで取り上げる問題は，囚人のジレンマゲームと呼ばれる非ゼロ和二人ゲームであり，次のように定式化される．参加者は，C(協調：Cooperate)，または D(裏切り：Defect) のいずれかの手を出すことができ，双方の手の組合せにより表 1 のような得点を得る．

表 1．対戦収益/損失表

| A \ B | C | D |
|---|---|---|
| C | A: 3 B: 3 | A: 0 B: 5 |
| D | A: 5 B: 0 | A: 1 B: 1 |

このゲームは，同じ相手と繰り返し対戦するという設定では，たとえ利己的なエージェントであっても協調が可能であるが，一回限りの対戦では，裏切りが唯一合理的な戦略であることが知られている．我々の目的は，過去の対戦履歴を公開することで，このジレンマ状況を打開しようというものである．

以下に，本論文に必要な範囲でモデルの概要を与える [Ito 95]．エージェントは，2次元空間上の「世界」をランダムに歩き回っており，同時に同じ場所にいるエージェント同士は「対戦」を行なうことができる．対戦は囚人のジレンマゲームと同型で，双方は協調 (C)，または裏切り (D) の手を出すことができる．このとき，エージェントがどの手を出すかは全くエージェントの自由である．ただし，対戦において双方が出した「手」は公表されて，記録として残される．また，このときの対戦履歴は，以後どのエージェントも自由に参照することができる．

エージェントは，最初に自己資産 $E = E_0$を持ち，対戦のたびに得られた利益／損出（表1から，対戦コストを引いたもの）が資産に加えられる．対戦コストは，固定の対戦手数料 $C_d$と，次に出すべき手を計算するに要したステップあたり $C_{am}$の計算コストからなる．

自己の資産が0になったエージェントは，死滅し系から削除される．逆に自己の資産が一定 ($E = 2E_0$) 以上になると，エージェントは自己の資産を分け与えて子エージェントを作る．子エージェントは親の対戦戦略を継承するが，対戦履歴は継承しない．戦略の継承に際しては，そのアルゴリズム長に応じた継承コストが子エージェントから徴収される．我々が採用した系のパラメタを表2に示す．

表2 系のパラメタ

| | |
|---|---|
| エージェント移動確率 $p_{rw}$ | 1/40 |
| 格子サイズ $N$ | 15 |
| 初期資産 $E_0$ | 20 |
| 対戦手数料 $C_d$ | 2.5 |
| 計算コスト $C_{am}$ | 0.002 |
| 戦略継承コスト $C_{st}$ | 0.1 |

## 3 対戦戦略の記述

対戦戦略の学習を議論するためには，戦略を具体的に記述する必要がある．そこで我々は，まず対戦戦略アルゴリズム計算器を定義し，その上で定義された命令セットを用いて，様々な対戦戦略を記述する [Ito 94]．計算器への入力は，(現在の) 時間，相手エージェントへのポインタ，及び戦略アルゴリズムであり，計算器は計算の結果として次に出すべき手 (C または D) を返す．この計算器は，8個の (型付きの) レジスタ (内部記憶) と，オペランドを持たない6個の命令セットを持つ．さらに，計算器は一個の無限長スタックを持つことで，再帰的な計算能力を備えている．我々は，エージェントが学習できる戦略を，本計算器で計算可能なものに制限する．

計算器への入力は，(現在の) 時間，相手エージェントへのポインタ，及び自己の対戦戦略アルゴリズムであり，計算器は計算の結果として次に出すべき手（C または D）を返す．計算器のレジスタ，命令セット，処理フローを C++の構文を用いて記述したものを図1に示す．

計算器は，相手の直前の対戦記録を history レジスタにロードする．次にアルゴリズムのコードを一つづつ読み込み，実行する．LC,LD は h(hand) レジスタにそれぞれ C，D をロードする．BM,BY,SL は分岐命令である．BM は相手エージェント自身の出した手，BY は直前の対戦で相手の相手が出した手，SL は直前の対戦で相手の立場にいたら自分が出したであろう手を調べて，その値が C か D かにより分岐をする．命令 TP は相手の対戦記録を一つ前に遡って，初めからアルゴリズムを適用し直す．処理はアルゴリズムが終端に達するか，または相手の対戦記録がなくなるかすれば終了し，その時の h レジスタの値を返す．

いくつかの基本的な対戦戦略を本命例セットで記述したものを以下に示す．

**1. お人好し戦略 (CCC)** ：(LC)
常に C を出し続ける．

**2. しっぺ返し戦略 (TFT)** ：(LC BM (LC) (LD))
相手の直前の手を出す．ただし，履歴がなければ C を出す．

**3. ランダム戦略 (RAN)** ：NIL
対戦相手に関係なく，1/2の確率で C と D とを選ぶ．

**4. 完全搾取戦略 (DDD)** ：(LD)
常に D を出し続ける．

**5. 修正しっぺ返し戦略 (TT3)** ：
(LC BM (LC) (BY (LD) (TP)))
相手の履歴から，双方とも D を出したものを除いて，相手の直前の手を出す．ただし，履歴がなければ C を出す．

**6. 自省戦略 (REF)** ：
(LC BM (LC) (BY (SL (LD) (TP)) (TP)))
相手の履歴から，双方とも D を出したものと，「自分でも同じ状況では D を出す」と思われるものを除いて，相手の直前の手を出す．ただし，履歴がなければ C を出す．

```
/* type definition */
enum hand {C,D};  // enumerate type {C,D}
typedef int time;  // time type
class agent;  // agent type
typedef int instruction-pointer ip;
struct register{  // registers
  hand h;  // output register
  time t;
  agent* a;
  history his;  // match history
  instruction-pointer ip;
};
struct history{
  time time;
  agent agent;  // the other agent
  hand my;      // hand of this agent
  hand you;     // hand of the other agent
};
class agent{ ... ;        // agent's private data
  history history;
};
/* opecode */
enum opcode {
    LC;  // loadh-c;  load C to h register
    LD;  // loadh-d;  load D to h register
    BM;  // branch-m; branch acc. his.my
    BY;  // branch-y; branch acc. his.you
    SL;  // apply self; apply its own algorithm
         //                          to his.agent
    TP;  // decriment time by 1, and goto top;
};
typedef al-codes List-of opcode;  // like LISP's list
/* abstract machine */
am(time t, agent a,  al-codes al) {
/* current time, the other agent, strategy algorithm */
  instruction-pointer ip=0;  /* set C or D randomly */
  hand h=random_selection('C','D');
/* load  agent(a)'s (at the time t) latest deal history to
                                          his registers  */
  history his=latest_rec(a,t);
  for(;;){  // fetch opecode
    switch (c=get_opcode(al,ip++)){
      case lc: h='c';braek;
       case ld: h='d';break;
       case bm: (his.my=='c')?
         ip=branch1, ip=branch2 ;break;
       case by: (his.you=='c')?
         ip=branch1, ip=branch2 ;break;
       case sl: (am(t-1, his.agent, al)=='c')?
         ip=branch1, ip=branch2 ;break;
       case tp: t--;ip=0;break;
     }
  }
  return h;
}
```

図1　対戦戦略アルゴリズム計算機

## 4　つき合い方戦略の学習

まず初めに，対戦履歴が公開されるという条件では，これまでの囚人のジレンマゲームと同じような簡単な戦略ではうまく行かないことを，確認しておく．対戦履歴が全て公開されるのだから，同じ相手と繰り返し対戦する「反復囚人のジレンマゲーム」と，そう大きく異ならないのではないかと思われるかも知れない．しかしながら，反復囚人のジレンマゲームでは，相手が先に裏切りを出してくれれば，非協調的なのは相手であり，自分ではないことは明らかである．ところが，我々のモデルの場合，相手の過去の行動履歴だけから，一体どのようにして相手の協調性を判断したら良いのだろうか．

しっぺ返しをとろうとしても，同じ相手とは繰り返し対戦しないのであるから，ふつうの意味でのしっぺ返しは不可能である．例えばあなたが旅先でお金を盗まれたからといって，翌日見ず知らずの人からお金を盗むことは何らしっぺ返しにはならない．また，別の人がお金を盗んだ犯人を見つけたとして，彼がその盗人のお金を巻き上げてしまったとしても，お金はあなたには戻ってこないわけで，これもしっぺ返しと呼べるかどうかは疑問である．

単純なしっぺ返しがそのままではうまくいかないことは，しっぺ返し戦略 (TFT) をその半数の完全搾取戦略（DDD）と対戦させてることで，実験でも確かめられた．我々のモデルでは対戦手数料が $C_d = 2.5$ なので，協調できない戦略は長期的に正の得点を稼げない．残念ながら TFT は，DDD との戦いの中で味方同士でも裏切りを出してしまい，同士討ちにより絶滅してしまうのである（図2参照）．

そこで，我々はしっぺ返し戦略 TFT をよりうまく協調できるように訓練することを考えた．そのためには，エージェントに何らかの学習能力が必要となるが，予め協調的性向を組み込んでしまっては面白くない．我々の採用した学習戦略は次のようなものである．現在の戦略を採用してからの自己の平均得点と，他のエージェントの平均得点とを観測しておく．もし，自分が他よりも良い得点を稼いでいるようであれば，危険を犯してまで自己の戦略を変更する必要はないであろう．しかしながら，逆に他よりも低い得点しか稼げていないとすると，自己の戦略を修正することを検討した方が良いであろう．この時，これまでの戦略と完全に無関係な戦略を採用することも可能であるが，これまでの経験を生かして，自己の戦略を少しだけ変更してみるというのが，妥当な学習則であろう．具体的な学習戦略を以下に

示す.

**戦略修正の契機** エージェントは，以下の状態変数を常に計測しておく.

$A(t) = 0.75A(t-1)$

$+0.25\times$ (全エージェントの時刻 $t$ での平均得点)

$N(t)$ =(現在の戦略を採用以後の自己の対戦回数)

$S(t)$ =(現在の戦略を採用以後の自己の得点の総和)

$\Delta(t) = A(t) - (S(t)/N(t) + \sigma/\sqrt{N(t)})$.

ここで右辺第二項は，$\sigma$ を自己の得点の標準偏差と考えると，対戦回数 $N(t)$，得点の総和 $S(t)$ から予想される得点平均値の上限である．従って，

$\Delta(t) \geq 0$

であれば，自己の得点が平均よりも明らかに低いことを意味し，戦略修正の契機となる．なお以下の計算では，$\sigma = 1.0$ を用いている.

**アルゴリズムの修正** 自己の戦略アルゴリズムが，戦略アルゴリズム計算器の長さ $len$ のコード列として表されているとする．この時，確率

$p_m = \mathrm{minimum}(10^{5\Delta(t)-2}, 0.4)/len$

を用いて，以下に示すようにアルゴリズムのランダムな修正を行なう.

1) コードの脱落 アルゴリズム中のそれぞれの命令コードが，確率 $p_m$ で脱落する.

2) コードの挿入 六個の命令セットからランダムに選ばれたコードが，それぞれの命令の間に確率 $p_m$ で挿入される.

このように，エージェントは自ら協調しようとか，非協調戦略に対抗しようというような，方向性を持った探索を一切行なっていない．唯一の学習戦略は，他と比較して自己の得点が低過ぎるようであれば，ランダムに自己の戦略を変更してみるというものである.

## 5 実験結果

まず最初に，このエージェントの社会は戦略学習のための動機 (刺激) を必要とする．実際，協調戦略だけの社会では，すべてのエージェントは単に協調 (C) を出すだけであり，エージェントにとって新しい戦略を学習する動機がない．そこで，非協調戦略のの集団 (以後ワクチンと呼ぶ) を人為的に系に注入することで，学習のための動機を作り出すこととした.

注入されたワクチンが強すぎると，他のエージェントはワクチンによって滅ぼされてしまう．逆に弱すぎると，それは学習のための刺激として有効に働かない．そこで，表2にあるように，DDD(常にDを出す)，またはRAN(ランダムにC，またはDを出す) からなるワクチンを少しずつ強度を増加させて，500ステップ毎に注入することとした．たとえば，時刻 [0, 10000] の区間では，全人口の10%のDDDワクチン (戦略DDDのエージェント) を500ステップ毎に注入したことを表す.

表2 訓練のために注入されたワクチン

| time | vaccine | vaccine ratio |
|---|---|---|
| 0-10000 | DDD | 10% |
| 10,000-20,000 | DDD | 20% |
| 20,000-30,000 | RAN | 20% |
| 30,000-40,000 | RAN | 33% |

学習の効果を評価するために，未学習のTFTの系，DDDによる訓練終了時 ($S_1$) の系，およびRANによる訓練終了時 ($S_2$) の系を，その人口の半数のDDD，または同数のRANと対戦させて，その振舞いを調べてみた．実験結果を図2（DDD対戦），図3（RAN対戦）に示す．未学習のTFTでは，非協調戦略との対戦によりほぼ全滅してしまったのに対して，学習後の系では一定時間後に人口が回復しており，より強い協調戦略を学習していることがわかる.

次に，どのような戦略が学習されたのかを見るため，$S_1$，$S_2$において，人口順に上位3位までの戦略を示したのが表3である．たとえば，$S_1$の1位の戦略はおおよそ次のように読むことができる.

$S_1$の1位 ：相手が最後にCを出していればCを出す．Dを出していれば，その時自分ならCを出すという状況であり，かつ相手の相手もCを出していればDを出す．それ以外では，相手の最後から2番めの履歴についてこの手順を適用する.

我々はこのような学習実験を5回行なった．面白いことに，どのような戦略が優勢になるかは，実験毎に異なっている．しかしながら，おおまかに言って $S_1$は対DDD戦闘能力を，$S_2$は対RAN戦闘能力を学習すること，自己のアルゴリズムを用いて相手を判断する再帰的戦略が，時間を追って優勢になることなどは共通しているようである.

## 6 考察とまとめ

我々は，利己的なエージェントのつき合い方，特に協調の問題を考えてみた．すなわち，自らの利益

をのみ追求する利己的なエージェントは，どのようにして他のエージェントや人間とつき合っていけば良いのであろうか．何の条件もなければ，おそらく協調は不可能であろう．では人は，また多くの生物たちはなぜ協調できるのであろうか．Dawkinsは遺伝子の共有を，またAxelrodは同じ相手との継続的つき合いにその理由を求めた．我々は第3の可能性として，過去の行動履歴の公開を取り上げ，利己的なエージェントがどのように行動すればよいかを調べてみた．その結果我々は次のことを明らかにした．

・適切な環境では利己的なエージェントも，自己の利益のために協調することができる．

・協調のためには，相手の過去の行動から，協調的かどうかを判定する能力が必要となる．

・エージェントは対戦の中で自己の利益を追求する過程で，そのような協調の仕方を学習する．

しかしながら，協調の仕方を学習するためには，何らかの刺激（非協調なエージェントの存在）が必要であった．この結果は，本質的である．平和な（協調的戦略ばかりの）社会では，有限の計算コストのため，何も考えずにCを出す戦略がもっとも有利である．従って,「強い」協調的戦略から出発した社会も，長い平和の時代が続くと各エージェントが「何も考えずにCを出す戦略」を学習し，非協調戦略に対抗する能力を失ってしまう．その結果は，ある時突然に進入した非協調戦略によって，協調的社会が敢えなく滅ぼされてしまうことになる．「平和は戦いの中でしか維持できない」という教訓は，我々のモデルにおいても，人間社会においても，忘れてはならない「真理」なのである．

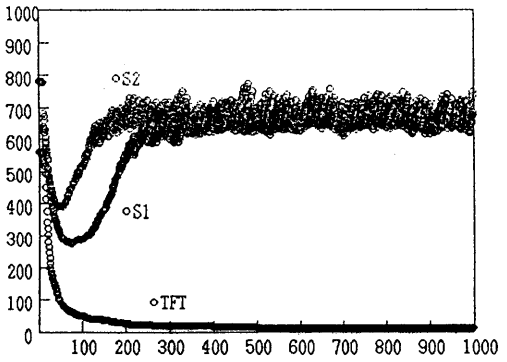


図2 TFT，$S_1$，$S_2$のDDD対戦での振舞い

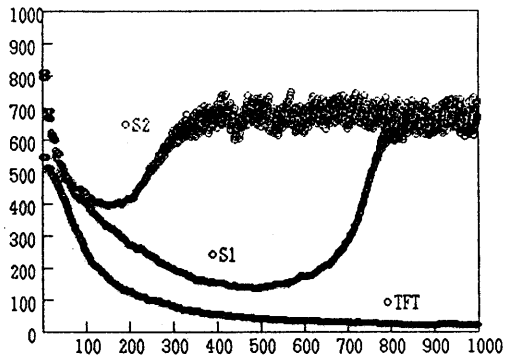


図3 TFT，$S_1$，$S_2$のRAN対戦での振舞い

表4 $S_1$，$S_2$上位3種の戦略

| rank | population | strategy |
|---|---|---|
| $S_1$ 1 | 11.7% | (BM (LC) (LD BY (SL NIL (LC TP)) (TP))) |
| $S_1$ 2 | 11.7% | (BM (LC) (LD BY (SL (LD) (LC TP)) NIL)) |
| $S_1$ 3 | 7.2% | (BM LC (BY (LD) (SL NIL NIL))) |
| $S_2$ 1 | 20.4% | (BM (LC BY NIL NIL) (LD BY (SL NIL (LC TP)) (TP))) |
| $S_2$ 2 | 8.1% | (BM (LC BY NIL (LC)) (LD BY (SL NIL (LC TP)) (TP))) |
| $S_2$ 3 | 6.6% | (BM (LC BY NIL (SL NIL NIL)) (LD BY (SL (LD) (LC TP)) (TP))) |

## 参考文献


[Axelrod 84] Axelrod,R.: *The Evolution of Cooperation*, Basic Books Inc., (1984)，松田裕之訳，つきあい方の科学，HBJ出版局，(1987).

[Dawkins 76] Dawkins, R: *The Selfish Gene*, Oxford: Oxford Univ. Press, 日高敏隆他訳，利己的な遺伝子，. 紀伊国屋書店, 1991.

[Ishida 95] 石田亨:「エージェントを考える」，人工知能学会誌，Vol.10, No.5, pp.663-667 (1995).

[Ito 94] 伊藤昭，矢野博之：「取引戦略の社会的進化」，MACC'94(第回マルチエージェントと協調計算ワークショップ) 予稿.

[Ito 95] 伊藤昭，矢野博之:「取引履歴公開下での最適取引戦略」，人工知能学会誌，Vol.10, No.2, pp.271-278 (1995).