人工知能 期末レポート

学籍番号：045713C
氏　名：大城和也

提出日：平成 18 年 2 月 7 日

1　課題内容

R.Axelrod の IPD コンテストに出場するための IPD ルールを設計し, 評価せよ. 利得テーブルは下記を使用するものとし, 基本ルールは第一回大会に準ずるものとする.

2　プログラム

今回は対戦させるための全体のプログラムを C 言語で作成したのでそれは巻末にて乗せる．以下は自作の IPD のオリジナルプログラムとそれを組み込んで動かした時の実行結果である.

2.1　ソース (IPD オリジナル)

```
-------------------------------------------------------------------------------
/* 前半は大体しっぺ返し，後半は裏切りが主で最後は必ず裏切る代物 */
int original(int round, int *own, int *enemy)
{
  if (round == 1) return (0);

  if (round > 1){
    if(round <= ROUND/2){ /* 前半 */
      if((enemy[round-1]==0))
return (round%2);  /* ラウンド数が奇数なら 1 を偶数なら 0 を返す */
      else return (1);
    }else if(round < ROUND){ /* 後半 */
      /* 自分対策 */
      if ((enemy[round-3]==own[round-3])&&(enemy[round-2]==own[round-2])&&
  (enemy[round-1]==own[round-1]))
return(0);
      else if((enemy[round-3]==1)&&(enemy[round-2]==1)&&(enemy[round-1]==1)){
return (1);
      }
      else if ((enemy[round-2]==0)&&(own[round-1]==0)){
return (1);
      }else return(round%2);
    }else return (1); /* 最後 */
  }
}
-------------------------------------------------------------------------------
```

このプログラムの動作は前半戦はほとんどしっぺ返しと同じような動作をする．この時の違いは相手が協調の場合，自分はラウンドが偶数か奇数かによって協調か裏切るかが変化する．そして，後半戦では履歴から自分同士で戦っている可能性も配慮して自分同士だと考えられる時には協調するようにした．その他は相手が裏切り続けているようなら自分も裏切る．それ以外 (最後を省く) ならまた，round によって変化する．そして最後は必ず裏切る.

2.2 実行結果

```
********************************************************************************
[nw0413:sophomore/AI/final-report] j04013% ./ipd_test
prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : distrust
Entry no.1

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : rd
Entry no.2

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : goodness
Entry no.3

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : tft
Entry no.4

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : repet
Entry no.5

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : hypocrisy
Entry no.6

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : all_c
Entry no.7

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : dwplus
Entry no.8

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
select the entry prisoners : original
Entry no.9

prisoners list-------------------------------------------------------
[distrust] [rd] [goodness] [tft] [repet] [hypocrisy] [all_c]
[dwplus] [original]
---------------------------------------------------------------------
```

```
select the entry prisoners :
  Entry No :  |  1| |  2| |  3| |  4| |  5| |  6| |  7| |  8| |  9|
    Points =  [ 20] [  0] [  0] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [ 56] [ 11] [  0] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [ 28] [  0] [ 18] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [ 24] [  0] [  0] [ 19] [  0] [  0] [  0] [  0] [  0]
    Points =  [ 60] [  0] [  0] [  0] [ 10] [  0] [  0] [  0] [  0]
    Points =  [ 96] [  0] [  0] [  0] [  0] [  1] [  0] [  0] [  0]
    Points =  [100] [  0] [  0] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [ 40] [  0] [  0] [  0] [  0] [  0] [  0] [ 15] [  0]
    Points =  [ 36] [  0] [  0] [  0] [  0] [  0] [  0] [  0] [ 16]
    Points =  [  0] [ 47] [  0] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [ 60] [ 40] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [ 45] [  0] [ 45] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [ 53] [  0] [  0] [ 28] [  0] [  0] [  0] [  0]
    Points =  [  0] [ 82] [  0] [  0] [  0] [ 22] [  0] [  0] [  0]
    Points =  [  0] [ 80] [  0] [  0] [  0] [  0] [ 30] [  0] [  0]
    Points =  [  0] [ 46] [  0] [  0] [  0] [  0] [  0] [ 41] [  0]
    Points =  [  0] [ 37] [  0] [  0] [  0] [  0] [  0] [  0] [ 57]
    Points =  [  0] [  0] [ 60] [  0] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [ 60] [ 60] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [ 30] [  0] [ 80] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [ 57] [  0] [  0] [ 62] [  0] [  0] [  0]
    Points =  [  0] [  0] [ 60] [  0] [  0] [  0] [ 60] [  0] [  0]
    Points =  [  0] [  0] [ 60] [  0] [  0] [  0] [  0] [ 60] [  0]
    Points =  [  0] [  0] [ 30] [  0] [  0] [  0] [  0] [  0] [ 80]
    Points =  [  0] [  0] [  0] [ 60] [  0] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [ 50] [ 50] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [ 57] [  0] [ 62] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [ 60] [  0] [  0] [ 60] [  0] [  0]
    Points =  [  0] [  0] [  0] [ 60] [  0] [  0] [  0] [ 60] [  0]
    Points =  [  0] [  0] [  0] [ 47] [  0] [  0] [  0] [  0] [ 52]
    Points =  [  0] [  0] [  0] [  0] [ 40] [  0] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [ 77] [ 32] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [ 80] [  0] [ 30] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [ 41] [  0] [  0] [ 41] [  0]
    Points =  [  0] [  0] [  0] [  0] [ 27] [  0] [  0] [  0] [ 52]
    Points =  [  0] [  0] [  0] [  0] [  0] [ 58] [  0] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [ 62] [ 57] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [ 62] [  0] [ 57] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [ 31] [  0] [  0] [ 76]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [ 60] [  0] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [ 60] [ 60] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [ 30] [  0] [ 80]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [  0] [ 60] [  0]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [  0] [ 30] [ 80]
    Points =  [  0] [  0] [  0] [  0] [  0] [  0] [  0] [  0] [ 42]
Sum Points = [ 460][ 461][ 415][ 458][ 433][ 392][ 387][ 424][ 535]
********************************************************************************
```

2.3 説明と考察

今回は多人数試合を行った．実行結果の下の表のようになっているものがその結果である．Entry No が書かれているがそれは左から

- distrust — 裏切り続ける
- rd — ランダム
- goodness — 初めの 2 手は協調，二度続けて裏切られたら裏切る
- tft — しっぺ返し
- repet — 協調と裏切りの交互

- hypocrisy — 最後だけ裏切る
- all_c — 協調し続ける
- dwplus — DOWNING の改良版
- original — 自分で作成したの

となっている．Points と書かれているのがそれぞれの試合での得点を表しており，Sum Points で全体の合計点を出している．今回の優勝は自分で作成したプログラム．

それぞれの試合での勝率は高く，distrust 以外には全て勝利している．tft にも 5 ポイント差で勝利している．これは最後に裏切る動作を加えたためである．自分同士での戦いでは 42 ポイント，自分自身との勝負かどうかを判断することを考慮しない場合には 24 ポイントであった．

ある程度強くなるためには相手の兆候を考えることと自分との勝負をどうするかを考える必要がある．tft は簡単でありながら，これら二つを考えている．よって，平均的に強いものには tft の形を継承しながら行動するようなものが多いと思われる．今回のプログラムもそれをイメージして作成した．

3 感想

今回は IPD のルールを考えることが主なものでしたが，自分はプログラムの方に力を入れてしまって … 。まぁ，それでもかなり中途半端な仕上がりになりましたが …．IPD のルールの方は，それなりに勝率のよさそうなものができましたが大体しっぺ返しと同じような動作をする物になりました．ルールが簡単なだけに考えるのが難しい物ですね．参考のイギリスの大学チームの話は面白い話でしたがやっぱ卑怯なイメージがついてきますね．かといって他に勝つ方法を考えろと言われると，いやはや奥が深いものです．

プログラムソース

```
--------------------------------------------------------------------------------
/* date   : 2006 年 2 月 3 日
 * author : Kazuya Oshiro
 * This program is test program for IPD
 */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define n_elements(a) (sizeof a / sizeof a[0])
#define ROUND 20

int distrust(int round, int *own_result, int *opposite_result);
int rd(int round, int *own_result, int *opposite_result);
int goodness(int round, int *own_result, int *opposite_result);
int tft(int round, int *own_result, int *opposite_result);
int repet(int round, int *own_result, int *opposite_result);
int hypocrisy(int round, int *own_result, int *opposite_result);
int all_c(int round, int *own_result, int *opposite_result);
int dwplus(int round, int *own_result, int *opposite_result);
int original(int round, int *own_result, int *opposite_resutl);
void entry();
void judge();


typedef int (*prisoner_t)(int round, int *own_result, int *opposite_result);

/*************************************************************************
 * 新しい IPD を参加させる場合には以下の prisoners と prisoners_name の後の値に  *
 * その関数名を最後に加えて下さい.                                          *
 *************************************************************************/
const prisoner_t prisoners[] = {distrust, rd, goodness, tft, repet, hypocrisy,
all_c, dwplus, original};
char *prisoners_name[] = { "distrust", "rd", "goodness", "tft", "repet",
   "hypocrisy", "all_c", "dwplus", "original"};
const int n_prisoners = n_elements(prisoners);
prisoner_t entry_pri[(sizeof prisoners)];
int n_entry;


int main()
{
  entry();
  judge();

  return (0);
}

void entry()
{
  int ei;
  n_entry=0;
  char e_prisoners[256];

  do {
    printf("prisoners list-------------------------------------------------------\n");
    for (ei=0; ei<n_prisoners; ei++){
      if(ei == 7) printf("\n");
      printf("[%s] ", prisoners_name[ei]);
    }
    printf("\n---------------------------------------------------------------------\n");

    printf("select the entry prisoners : ");
    fgets(e_prisoners, 256, stdin);

    e_prisoners[strlen(e_prisoners)-1] = '\0';
    for(ei=0; ei < n_prisoners; ei++)
      if (strcmp(e_prisoners,prisoners_name[ei]) == 0 ){
entry_pri[n_entry] = prisoners[ei];
n_entry++;
printf("Entry no.%d\n\n", n_entry);
```

```
      }
  }while((strcmp(e_prisoners,"\0") != 0));
}

void judge()
{
  int a=0, b=0;
  int round;
  int i, j, n,  p[n_entry], p_out[n_entry];
  int result_r[n_entry][ROUND];
  char join[256];

  printf("  Entry No : ");
  for(n=0; n<n_entry; n++)
    printf(" |%3d|", n+1);
  printf("\n");

  for(i=0; i<n_entry; i++)
    p_out[i] = 0; /* 値の初期化 */

  for(i=0; i<=(n_entry-1); i++){ /* 組み合わせでの対戦 */
    for(j=i; j<=(n_entry-1); j++){
      for(n=0; n<n_entry; n++)
      p[n] = 0; /* 値の初期化 */
      for(round=1; round<=ROUND; round++){
a = entry_pri[i](round, result_r[i], result_r[j]);
b = entry_pri[j](round, result_r[j], result_r[i]);

if( (a==0) && (b==0) ) { p[i]+=3; p[j]+=3; } /* 協調：協調 */
if( (a==1) && (b==0) ) { p[i]+=5; p[j]+=0; } /* 裏切り：協調 */
if( (a==0) && (b==1) ) { p[i]+=0; p[j]+=5; } /* 協調：裏切り */
if( (a==1) && (b==1) ) { p[i]+=1; p[j]+=1; } /* 裏切り：裏切り */

result_r[i][round] = a;
result_r[j][round] = b;
// printf("%d:%d\n", p[i], p[j]);
      }
      if(i==j){p[i] = p[i]/2; p_out[i] -= p[i];}

      printf("    Points = ");
      for(n=0; n<n_entry; n++)
printf(" [%3d]", p[n]);
      printf("\n");

      p_out[i] += p[i];
      p_out[j] += p[j];
    }
  }
  printf("Sum Points = ");
  for(i=0; i<n_entry; i++)
    printf("[%4d]", p_out[i]);
  printf("\n");
}

//全部協調
int all_c(int round, int *own, int *enemy)
{
  return (0);
}

//最初の 2 手は協調．その後，二度続けて裏切られたら裏切り返す．
int goodness(int round, int *own, int *enemy)
{
  if(round==1 || round==2) return (0);
  if(enemy[round-1]==1 && enemy[round-2]==1)
    return (1);
  else
    return (0);
}

//ランダム
int rd(int round, int *own, int *enemy)
{
  return randam();
```

```
}

//すべて裏切る
int distrust(int round, int *own, int *enemy)
{
  return (1);
}

//協調と裏切りの繰り返し
int repet(int round, int *own, int *enemy)
{
  if(round%2 == 0) return (0);
  else return (1);
}

//しっぺ返し
int tft(int round, int *own, int *enemy)
{
  if (round == 1) return (0);
  if(enemy[round-1] == 1) return (1);
  else return (0);
}

//最後だけ裏切る
int hypocrisy(int round, int *own, int *enemy)
{
  if(round == ROUND) return(1);
  else return(0);
}

//DOWNING の改良版
int dwplus(int round, int *own, int *enemy)
{
  static int p,q, flag, count_d;

  if ( round == 1 ){
    p=q=0;
    flag=0;
    count_d=0;
    return 0;
  }

  if (round > 1 ){
    if((enemy[round-1]==0)&&(own[round-1]==0)){
      p++;
    }else if ((enemy[round-1]==1)&&(own[round-1]==1)){
      q++;
    }

    if ( flag ) return (1);
    if((round > 1)&&(round < 11)&&(enemy[round-1]==1)){
      count_d++;
      if((round == 10)&&(count_d>8))flag=-1;
    }

    if ( p==q ){
      return (1);
    }else if( p>q ){
      return (0);
    }else{
      return round%2;
    }
  }
}

/* 前半は大体しっぺ返し，後半は裏切りが主で最後は必ず裏切る代物 */
int original(int round, int *own, int *enemy)
{
  if (round == 1) return (0);

  if (round > 1){
    if(round <= ROUND/2){ /* 前半 */
      if((enemy[round-1]==0))
return (round%2);  /* ラウンド数が奇数なら 1 を偶数なら 0 を返す */
```

```
      else return (1);
    }else if(round < ROUND){ /* 後半 */
      if ((enemy[round-3]==own[round-3])&&(enemy[round-2]==own[round-2])&&
  (enemy[round-1]==own[round-1]))
return(0);              /* 自分対策 */
      else if((enemy[round-3]==1)&&(enemy[round-2]==1)&&(enemy[round-1]==1)){
return (1);
      }
      else if ((enemy[round-2]==0)&&(own[round-1]==0)){
return (1);
      }else return(round%2);
    }else return (1); /* 最後 */
  }
}

//乱数作成
int randam()
{
  static int num;
  if(num == 0){
    srand((unsigned int )time(0));
    num = 1;
  }
  return (int)(rand()*2.0/(1.0+RAND_MAX));
}
```

参考文献

[1] 【開催】囚人のジレンマ大会”http://pc.2ch.net/tech/kako/1019/10190/1019033323.html”

[2] IPD - Home page ”http://www.lifl.fr/IPD/ipd.frame.html.en”

[3] 先輩方々の過去のレポート