PATTERNS IN SCISSORS, PAPER AND ROCK

BY

JOHN PRICE* AND MATTHEW PRICE**

The game Scissors, Paper and Rock (SPR) has a long history. One famous contest took place in the pages of *You Only Live Twice* by Ian Fleming [2] and resulted in our hero James Bond defeating the inscrutable Tiger Tanaka.

Each round of the game consists of the two players simultaneously choosing one of scissors, paper or rock. The winner is decided by the rules "scissors cuts paper", "paper covers rock" and "rock blunts scissors". If both players make the same selection, then it is a draw. One of the attractions of the game are the hand signals to indicate each person's choice.

If a player randomly selects between the three possibilities, then clearly the scores will turn out evenly in the long run no matter what the opponent does. So what is the problem? For a game as simple as SPR, any problems are going to be extremely mild. Nevertheless, we can detect two, one mathematical and one behavioural. The mathematical problem is that, without recourse to a random number generator (such as a table of random numbers) or at least a pencil and paper, it is difficult to keep making random selections. The second problem is that few people are satisfied with coming out even "in the long run"; most want to outwit or outguess their opponents. To do this, players have to home in on any conscious or unconscious patterns of play of their opponents.

Of course, few of us have the insight of James Bond. The purpose of the following is to overcome this lack by introducing you to a way of writing a program to search out patterns of past play in this type of game, and then to choose accordingly. To make the project more interesting, let us consider a game which has the same problems as SPR but which also has a few more subtleties. The game is called Undercut and was devised by Robert Boeninger and Douglas Hofstadter and described by the latter in Scientific American [3].

---

* *John Price is a Senior Lecturer in the School of Mathematics at the University of New South Wales.*

** *Matthew Price is a Form 4 student at Sydney Grammar School.*

It is a game for two players with each round consisting of the simultaneous selection of integers in the range 1 to 5. The players score the number they choose except when they differ by precisely 1. In this case the player with the lower number scores his opponent's number as well as his own, while his opponent scores nil. There are two obvious, but opposing, strategies. Choose mostly high numbers (and not worry about being "undercut") or choose mostly low numbers (and hope that this is compensated for by frequent undercuts).

In practice players try to bluff and outguess their opponents. For example, player A might choose 5 a few times in a row to lure player B into choosing a 4. A's plan is to switch to a 3 in the same round as B chooses a 4. But which round is this? In any case, B might guess that this is A's strategy and plans to choose a 2 in the crucial round. Of course, there is a limit to the number of levels to which this guessing and double-guessing can be taken.

One thing is certain, if player A chooses the numbers at random, then he will be defeated by B continually playing a 4. (Why?) In a future article we will look more closely at strategies based on random selections which are weighted so that, in the long run, the number of 1's equals a fixed proportion of the number of 2's, and so on. Here we present a simple program written in Applesoft BASIC which allows you to play against the computer.

The program records the last moves of you and the computer, then looks over all the past occurrences of this pattern and determines which is your most common subsequent move. It makes its selection accordingly. The past moves are also weighted so that the more recent occurrences of a particular pattern are more significant. For example, suppose that the last moves by you and the computer are 3 and 5 respectively. The computer looks at the past occurrences of this pattern, sees that you usually follow this with a 4 and so responds with a 3.

Other types of patterns can also be used by the computer as a basis for a strategy. For example, it may base its response on your last two moves instead of the moves described above. The following program is easily modified to do this. Try it, and then pit the two programs against each other.

In [3], Hofstadter mentions that he wrote a program to play Undercut. Earlier versions for a similar game were also implemented at Bell Laboratories [1, p.89]. A general article which will help you to understand further the main results and subtleties of the mathematical theory of game theory is [4]. A more advanced treatment is given in [1], where strategies for a wide variety of games, including roulette, poker, blackjack and bridge, are examined.

1. R.A. Epstein, *The Theory of Gambling and Statistical Logic* (Academic Press, New York, 1977).

2. Ian Fleming, *You Only Live Twice* (New American Library, New York, 1964 (fifth printing)).

3. D.R. Hofstadter, *Metamagical themas* (Scientific American, August, 1982, pp. 10-14).

4. A. Rapoport, *The use and misuse of game theory* (in *Mathematics in the Modern World* (Freeman, San Francisco, 1968).

UNDERCUT PROGRAM

Lines 1 to 4 and 10,000 only concern the introduction and can be deleted. Also, if you prefer the computer to base its predictions on your last two moves, lines 300 and 1110 should be altered as follows:

```
300  IF  R > 1  THEN  PX(P1(R - 1),  P1(R - 2),  P1(R)))
     = PX(P1(R - 1),  P1(R - 2),  P1(R) + 1.05 ^ R)

1110 LM = P1(R - 1) : HM = P1(R - 2) : B = 0: etc.
```

The full program is:

```
1  HOME :A$ = "UNDERCUT ":B$ =  "]":C$  =  CHR$  (91):D$  =  "BY
M.PRICE":E$ = "PRESS ANY KEY TO CONT.": FOR X = 1 TO  LEN  (A$):
VTAB 3: PRINT  RIGHT$ (A$,X): GOSUB 10000: NEXT : FOR X  =  1  TO
16: VTAB 3: HTAB X: PRINT " ";A$: GOSUB 10000: NEXT
2 DX = 1:DY = 1:XX = 2:YY = 15:OY = YY: FOR X = 1  TO  171:  VTAB
OY: PRINT  TAB( 41);: VTAB YY: HTAB XX: PRINT B$;: HTAB 41 -  XX:
PRINT C$: IF XX = 39 OR XX = 1 THEN DX =  - DX
3  IF YY = 23 OR YY = 5 THEN DY =  - DY
4 XX = XX +  DX:OY = YY:YY = YY +  DY: NEXT : POKE 34,5: VTAB 23:
HTAB 16: PRINT D$: FOR X = 1 TO 16: CALL  -  912:  NEXT  :  POKE
34,0: GOSUB 10000: VTAB 15: FOR X =  1  TO  11:  HTAB  10:  PRINT
LEFT$ (E$,X); RIGHT$ (E$,X);: NEXT : WAIT  - 16384,128: GET A$
5  HOME : VTAB 3: PRINT  TAB( 15);"UNDERCUT ]"; CHR$ (91)
6  VTAB 6: PRINT "I HAVE MADE MY SELECTION FOR ROUND "
7  VTAB 8: PRINT  TAB( 3);"NOW YOU CHOOSE A NUMBER IN THE": PRINT
: PRINT  TAB( 9);"RANGE ONE TO FIVE."
8  VTAB 15: PRINT  TAB( 5);"YOUR CHOICE"; TAB( 25);"MY CHOICE"
10  DIM P1(200),P2(200),PX(5,5,5)
11  REM  P1(X) IS YOUR MOVE FOR ROUND X
12  REM  P2(X) IS MY MOVE FOR ROUND X
20  VTAB 13: PRINT "SCORE#1        ROUND#        SCORE#2"
99  REM  R REPRESENTS THE ROUND NUMBER
100  FOR R = 1 TO 200: VTAB 6: HTAB 36: PRINT R;"."
110  GOSUB 1100:P2(R) = A
120  GOSUB 1000:P1(R) = A
200  IF P1(R) + 1 = P2(R) THEN S1 = S1 + P1(R) + P2(R): GOTO 230
201  REM  S1 IS YOUR SCORE
202  REM  S2 IS MY SCORE
210  IF P2(R) + 1 = P1(R) THEN S2 = S2 + P1(R) + P2(R): GOTO 230
220 S1 = S1 + P1(R):S2 = S2 + P2(R)
230  VTAB 17: HTAB 10: PRINT P1(R);: HTAB 30: PRINT P2(R)
240  VTAB 13: HTAB 9: IF S1 > S2 THEN  INVERSE
250  PRINT S1;: NORMAL : HTAB 22: PRINT  R;:  IF  S2  >  S1  THEN
INVERSE :
260  HTAB 36: PRINT S2: NORMAL
300  IF R > 1 THEN PX(P1(R -  1),P2(R -  1),P1(R))  =  PX(P1(R  -
1),P2(R - 1),P1(R)) + 1.05 ^ R
310  NEXT : END
1000  WAIT  -  16384,128: GET A$:A =  VAL (A$): IF A < 1 OR A > 5
THEN 1000
1010  RETURN
1100  IF R < 9 THEN  GOSUB 1200: RETURN
1110 LM = P1(R - 1):HM = P2(R -  1):B = 0: FOR X = 1  TO  5:  IF
PX(LM,HM,X) > B THEN B = PX(LM,HM,X):A = X
1120  NEXT :A = A - 1: IF A < 1 THEN A = 4
1130  RETURN
1200 A =  INT ( RND (4) * 5) + 1: RETURN
10000  FOR Y = 1 TO 100: NEXT : RETURN
```