Anti-Mind, *Mastermind with Feedback* and *Anti-Mind with Lies*: Rediscovering the Strong AI Project through Variations of Mastermind

J. Barahona da Fonseca
jbfo@fct.unl.pt
Department of Electrical Engineering and Computer Science
Faculty of Sciences and Technology of New University of Lisbon
Monte da Caparica, 2829-516 Caparica, Portugal
www.dee.fct.unl.pt

When I started my career, in 1986, my main research interests were Artificial Intelligence(AI), Expert Systems and Decision Support Systems based on AI tools. The first experiment that I have done was the development of the *Anti-Mind* and *Master Mind with Feedback* programs written in the Basic language [1]. The *Anti-Mind* program simulates a *good* player of the Master Mind game, discovering the secret code defined by the human operator (a sequence of numbers in a pre-defined interval) very quickly. Then I used the algorithm of *Anti-Mind* to help and correct a human operator trying to discover the secret code defined by the computer resulting in the *Master Mind with Feedback*. Recently I revisited this work [2] and since then I worked the solution of Mastermind with an unlimited number of lies or errors. It seems that I developed a humanoid that thinks better than the human. Let's take an example to clarify what I mean by the 'Computer *Thinks* better than the human':
Anti-Mind Program , CPC=Number of Correct Digits in Correct Position
CPE=Number of Correct Digits in Incorrect Position, 3 Digits, Interval [0,3]
103 CPC,CPE=1,1 132 CPC,CPE=1,1 120 CPC,CPE=0,2 **Enough Information!**, Secret code=? The computer knows that the information is enough and it also knows the secret code. And you? In this paper I will present the algorithm of *Anti-Mind with an unlimited number of lies* which detects and identifies all the moves that are lies or errors, which is based on the Anti-Mind algorithm and I will present an example with 6 lies and a secret code with 4 digits varying between 0 and 9. I will discuss, at the light of Cognitive Science, why no human is capable of doing this. Finally I will propose a neural network architecture and a learning algorithm based on a hybrid reinforcement algorithm to model the learning process of a human interacting with *Mastermind with Feedback*. The detailed definition of this network for normal and subjects with psychological and psychiatric disorders will help to understand how they affect the performance in learning processes of complex tasks like learning to *play well Master Mind*.

Keywords: AI, Mastermind with Lies, Cognitive Science, Simulation of Human Behavior, Neural Network to Learn to Play Mastermind

1.Introduction

We all have the idea that the Brain is a very powerful logic processor. I will show the contrary: the Brain is a very weak logic processor, and we have a great difficulty to combine (equivalent to logical conjunction AND) various incomplete informations like in the Master Mind game. The problem is that, if we don't repeat the digits in the first moves, the logical expressions that represent the possible hypotheses coherent with each move are more complex and much more their conjunction. For example, if the first move is 1333 cpc=1 cpe=0, the logical expression of the possible hypotheses coherent with this information is

(1,1)&(3,de) ⊕ (1,de) & [(3,2)⊕(3,3)⊕(3,4)] & (3 doesn't exist in position 1)

where ⊕ represents the exclusive or (XOR) logical operation and (i,j) means *digit i exists in position j* and (i,de) means *digit i doesn't exist*.

But if the first move is

0254 cpc=1 cpe=2, the logical expression of the possible hypotheses coherent with this information is much more complex (assuming a maximum digit of 6):

(0,1)& { (2,3)&(5,2)&(4,de)&[(3,4) ⊕(1,4) ⊕(2,4) ⊕(5,4) ⊕(6,4)] ⊕(2,3)&(5,4)&(4,de) &[(3,2) ⊕(1,2) ⊕(5,4) ⊕(6,4)] ⊕ (2,4)&(5,2)&(4,de)&[(3,3) ⊕(1,3) ⊕(5,3) ⊕(6,3)] ⊕ (2,3)&(4,2)&(5,de)&[(1,4) ⊕(3,4) ⊕(4,4) ⊕(6,4)] ⊕ (2,4)&(4,2)&(5,de)&[(1,3) ⊕(2,3) ⊕(4,3) ⊕(6,3)] ⊕(2,4)&(4,3)&(5,de)&[(1,2) ⊕(2,2) ⊕(3,2) ⊕(4,2) ⊕(6,2)] ⊕(5,2)&(4,3)&(2,de)&[(1,4) ⊕(3,4) ⊕(4,4) ⊕(5,4) ⊕(6,4)] ⊕(5,4)&(4,2)&(2,de)&[(1,3) ⊕(3,3) ⊕(4,3) ⊕(5,3) ⊕(6,3)] ⊕(5,4)&(4,3)&(2,de)&[(1,2) ⊕ (3,2) ⊕ (4,2) ⊕(5,2) ⊕(6,2) } ⊕ (2,2)& {…} ⊕ (5,3)& {…} ⊕ (4,4)& {…}

note that A⊕B=A&not(B) + not(A)&B**,** where + means logical OR.

Now imagine the conjunction of various expressions like this…only a very powerful logic processor would be capable to make the conjunction (logical AND) of various expressions so complex without getting lost…as it happens with us when we try to understand how the anti-mind algorithm reaches the conclusion that the information is enough and finds the secret code. I have rewritten the anti-mind and master mind with feedback in matlab with some minor modifications in the algorithm and some more profound modifications in the implementation. Since the mathematical analysis of the worst-case performance of the anti-mind algorithm in terms of the maximum number of moves for each combination of number of digits and maximum digit (assuming that the digit varies between 0 and digit_max) is very complex due to the random nature of my anti-mind, I have also made two programs anti_mind_auto.m and anti_mind_auto2.m that put the computer playing against itself for each possible secret code, selecting the more unfavourable situations of enough information.

I made some simulations with these later programs but the results are not *reliable* since I used a number of repetitions with each secret code relatively small compared to the great number of possible combinations of good moves. For example [3] guaranteed that their anti-mind algorithm finds the secret code of 4 digits between 1 and 6 in no more than 9 moves, and my simulation points at a better worst case performance of 7 moves…but that doesn't mean that my algorithm is better, only that I didn't have access to a super computer!…and Donald E. Knuth[10} claims, without

proof, that his strategy guarantees a maximum of 5 moves! He showed that the best first guess is 1122. Knuth only solved the classic Mastermind game with 4 digits varying between 1 and 6 and didn't try to generalize or solve the same problem for a greater number of digits and/or maximum digit. It seems [10] was the first published work about the solution of the classic Mastermind game and the invention of the Master Mind game is attributed to M. Meyerowitz [17]. Since then many proposals were published [4,5,6,7,8,9,11,12,13,14,15], but no one did beat the worst case performance of Knuth's strategy.

2.Mastermind with Lies

Before presenting my Anti-Mind with lies algorithm, which finds any secret code with an unlimited number of lies, let's see an example with 6 lies, 4 digits varying between 0 and 9 and secret code 1559:

```
>> anti_mind_w_n_lies(4,9,0,1)
Number of Hypothesis=10000
Move 1=9  2  5  4, cpc=0, cpe=3, Number of Hypothesis=312
Move 2=5  9  8  2, cpc=0, cpe=2, Number of Good Hypothesis=93
Move 3=4  5  6  9, cpc=3, cpe=0, Number of Good Hypothesis=6
Move 4=4  5  3  9, cpc=3, cpe=0, Number of Good Hypothesis=5
Move 5=4  5  0  9, cpc=3, cpe=0, Number of Good Hypothesis=4
Move 6=4  5  7  9, cpc=2, cpe=0, Number of Good Hypothesis=0
**You Said one or more Lies...and Later On I will Tell you Where!!**
Number of Hypothesis=4
Move 7=4  5  4  9, cpc=2, cpe=0, Number of Good Hypothesis=0
**You Said More than 1 Lies...But I will Find Them!!**
Number of Hypothesis=4
Move 8=4  5  7  9, cpc=3, cpe=0, Number of Good Hypothesis=3
Move 9=4  5  7  9, cpc=2, cpe=0, Number of Good Hypothesis=0
**You Said More than 2 Lies...But I will Find Them!!**
Number of Hypothesis=3
Move 10=4  5  9  9, cpc=3, cpe=0, Number of Hypothesis=2
Move 11=4  5  4  9, cpc=2, cpe=0, Number of Good Hypothesis=0
**You Said More than 3 Lies...But I will Find Them!!**
Number of Hypothesis=2
Move 12=4  5  1  9, cpc=2, cpe=1, Number of Good Hypothesis=0
**You Said More than 4 Lies...But I will Find Them!!**
Number of Hypothesis=1
Move 13=1  5  9  9, cpc=3, cpe=0, Number of Good Hypothesis=0
Number of Hypothesis=1
Move 14=1  5  0  9, cpc=3, cpe=0, Number of Good Hypothesis=0
Number of Hypothesis=1
Move 15=1  5  3  9, cpc=3, cpe=0, Number of Good Hypothesis=0
Number of Hypothesis=1
Move 16=1  5  6  9, cpc=3, cpe=0, Number of Good Hypothesis=0
```

You Said More than 5 Lies...But I will Find Them!!
Number of Hypothesis=1
Move 17=1 5 5 9, cpc=4, cpe=0
**You Lied 6 Times in Moves:
** 1 3 4 5 8 10 **

After this digression you must be guessing the algorithm of Anti-Mind with lies, which I present next in pseudo-code:

```
function anti_mind_w_n_lies(n_digits,dig_max,flag_trace, flag_n_h)
[flag_lies anterior_moves]= anti_mind(n_digits, dig_max, flag_trace, flag_n_h)
if flag_lies
  disp('**You Said one or more Lies...and Later On I will Tell you Where!!**')
else
return
n_lies=0
n1=anterior_moves(1,1)
cf=(dig_max+1)^n_digits
while flag_lies
    n2=moves_ant(1,1)+1
    n_lies=n_lies+1
    if n_lies < (n1+1)
    n_lies_n1_l=n_lies
    else
    n_lies_n1_l=n1
    end
   % varying the number of lies in n1
    for n_lies_n1=n_lies_n1_l:-1:1
       % Inicalizing lie(i)
    for i=1:n_lies_n1
       lie(i)=i+1;
    end
    for i=n_lies_n1+1:n_lies
       lie(i)=n1+ i-n_lies_n1+1;
    end
       flag_limite=1;
       while flag_limite*flag_lies
       % regenerate good_moves assuming moves lie-1 as lies
       new_combination=zeros(1,n_digits);
       cardinal_g_m=0;
       for c=1:cf
       flag_coer=1;
       for j_a=2:moves_ant(1,1)+1
       j_a=act_j_a(j_a,lie,n_lies);
        if j_a > moves_ant(1,1)+1
          break
       end
```

```
    % calculating cpc_i and cpe_i  resulting from the comparison between
    % new_combination and moves_ant(j_a,1:n_digits)
    flag_coer=  flag_coer *
    (cpc_i==moves_ant(j_a,n_digits+1))*(cpe_i==moves_ant(j_a,n_digits+2));
    end % of for j_a=..
    if flag_coer
       cardinal_g_m=cardinal_g_m + 1;
        good_moves(cardinal_g_m,1:n_digits)=
        new_combination(1: n_digits)
    end
    end % of the for c=...
   flag_lies=(cardinal_g_m==0)
   if (1-flag_lies)
   moves_ant=anti_mind6(good_moves, cardinal_g_m, moves_ant, flag_trace,
flag_n_h);
   flag_lies=moves_ant(1,2)
   end
% generating new combination of possible lies
if flag_lies
 lie=gerar_nova_comb_ment(n_lies,n_lies_n1,lie,n1,n2)
 flag_limite=act_flag_limite(lie,n_lies_n1,n_lies,n1,n2)
end
end % of the while flag_limite*flag_lies
if (1-flag_lies)
   break
end
end % of the for n_lies_n1=...
if flag_lies
   disp(['**You Said More than ' int2str(n_lies) ' Lies...But I will Find Them!!**']);
end
end % of the while flag_lies
disp(['**You Lied ' integer2string(n_lies) ' Times in Moves:']);
disp(['** ' integer2string(lie-1) ' **']);
return
```

3.Neural Net that Learns to Play *Well* Mastermind

I'm just beginning the design of a neural net that learns to play *well* Mastermind. The net would have two main modules. The first one *stores* the previous guesses and respective cpc's and cpe's and the second generates the next guess that, when the net is well trained, would be always *coherent* with the previous guesses and respective cpc's and cpe's, that is, the final objective is that the net will always generate *good* moves. This second module would interact with my *Mastermind with Feedback* and the *feedback* (good or bad move) would be used by an hybrid reinforcement learning algorithm. Learning to play *well* Mastermind with lies is a much more difficult task and I would not consider this problem in the near future.

4. Conclusions and Future Work

Although a very simple game, mastermind puts in evidence human cognitive limitations [16]. My Anti-Mind algorithm seems to show a performance better than the performance of the best human Mastermind players. To clarify this question I'm planning to make experiments with a sample of very good human master mind players with anti_mind_real.m, that is, Mastermind with feedback. I will also explore the Anti-Mind with lies as a training program for criminal investigators and lawyers since they work with noisy data and data with lies.

References

[1]Barahona da Fonseca, J, *Curriculum Vitae*, (written in Portuguese), May 1989.
[2]Barahona da Fonseca, J, "*Anti-Mind* and *Master Mind with Feedback*: an Example where the Computer 'Thinks' better than the Human", in *Proceedings of Congresso em Neurociências Cognitivas*, November of 2003, in Press, University of Evora, Evora, Portugal.
[3]Charlton, G, Harrison, M and Jones, D, *The Turing Criterion : Machine Intelligent Programs for the 16K ZX81*, Interface Publications, 1982.
[4]Chvátal, V, "Mastermind", *Combinatorica*, 3:325-329, 1983.
[5] Flood, M M, "Mastermind Strategy", *Journal of Recreational Mathematics*, 18(3):194-202, 1985-86.
[6]Flood, M M, "Sequential Search Sequences with Mastermind Variants- part 1", *Journal of Recreational Mathematics*, 20(2):105-126, 1988.
[7]Flood, M M, "Sequential Search Sequences with Mastermind Variants- part 2", *Journal of Recreational Mathematics*, 20(3):168-181, 1988.
[8]Irving, R W, "Towards an Optimum Mastermind Strategy", *Journal of. Recreational Mathematics*, 11(2):81-87, 1978-79.
[9]Koyoma, K and Lai, T W, "An Optimal Mastermind Strategy", *Journal of. Recreational Mathematics*, 25(4):251-256, 1993.
[10]Knuth, D E, "The Computer as Master Mind", *Journal of. Recreational Mathematics*, Vol. 9(1), 1976, pp. 1-6.
[11]Neuwirth, E, "Some Strategies for Mastermind", *Zeitschrift fürOperations Research*, 26:B257-B278, 1982.
[12]Rada, R, "Mastermind in SIGART", *SIGART Newsletter*, 89:24-25, July 1984.
[13]Rao, T M, Kazin, G and O'Brien, D, "Algorithms to Play Mastermind", *SIGART Newsletter*, 95:33-35, January 1986.
[14]Shapiro, E, "Playing Mastermind Logically", *SIGART Newsletter*, 85:28-29, 1983.
[15]Sterling, L and Shapiro, E, *The Art of Prolog: Advanced Programming Techniques*", MIT Press, Cambridge, Massachusetts, second edition, 1994.
[16]Stillings, N A, Weisler, S E, Chase, C H, Feinstein, M H, Garfield, J L and Rissland, E L, *Cognitive Science: An Introduction*, second edition, MIT Press, 1995.
[17]N.I., "Games Gift Guide", *Games and Puzzles*, 20, pp. 16-17, December 1973.