ELSEVIER

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE @ DIRECT®
Theoretical Computer Science 313 (2004) 315–324

Theoretical Computer Science

www.elsevier.com/locate/tcs

Improved game play by multiple computer hints

Ingo Althöfer

Institut für Angewandte Mathematik, Friedrich-Schiller-Universität Jena, 07740 Jena, Germany

Received 26 April 2002; accepted 18 August 2003

Abstract

Humans and Computers have different strengths, which should be combined, not wasted. Multiple-choice systems are an efficient way to achieve this goal. An instructive example is 3-Hirn, where two programs make one proposal each and a human boss has the final choice amongst them. In Chess and several other brain games many experiments with 3-Hirn and other settings proved the usefulness of multiple choice systems and the validity of the underlying ideas.

Keywords: Multiple-choice system; 3-Hirn; Chess; Clobber; Combinatorial Game Theory; Zillions of Games

1. Introduction

Humans are able to think, to feel, and to sense. We can also compute, but not too well. In contrast, computers are giants in computing—they crunsh bits and bytes like maniacs. However, they cannot do anything else but computing. By combining the gifts and strengths of humans and machines in appropriate ways it is possible to achieve impressive results.

One such way is a “multiple choice system” [5]: One or several programs compute a clear handful of candidate solutions; then a human has the final choice amongst these candidates.

Brain games are ideal testbeds for decision support systems. Performance in play against other entities is a direct indicator of quality. So it is natural to test multiple-choice systems in game playing. In Section 2 we tell a story of success in Chess. First experiences with multiple-choice systems in the new game Clobber are described in Section 3. In Section 3 we conclude with a short discussion.

E-mail address: althofer@minet.uni-jena.de (I. Althöfer).

0304-3975/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2003.08.012

2. The 3-Hirn and other multiple-choice systems in Chess

In Chess, performance is internationally measured by the Elo-rating numbers. The stronger a player the higher his or her rating. If players A and B play against each other the difference $\mathrm{Elo}(A)-\mathrm{Elo}(B)$ allows to make a statistical prediction for the outcome. For instance, players with the same rating should have equal chances. If the difference is 200 Elo-points, the stronger player should make about 75% of the points.

Starting in 1985, the author ($\mathrm{Elo}=1900$) performed several experiments with multiple-choice systems in Chess [1,4,8]. We first describe the constructions used.

"*3-Hirn*" ('*Triple Brain*' *in English*): Two independent Chess programs propose one candidate move each. A human has the final choice amongst these candidates. If both moves coincide the human has to realize this single candidate. The human is not allowed to outvote the programs.

"*Double-Fritz with Boss*": In 1995, "Fritz" was the first Chess program for the mass market which had a k-best mode. (In k-best mode not only the single best but the k best moves are computed. Here k is some natural number.) In Double-Fritz with Boss Fritz was used in 2-best mode. The human is the boss and has the final choice amongst these two Fritz candidates.

"*List-3-Hirn*": This combines the concepts of 3-Hirn and Double-Fritz with Boss. Two independent Chess programs are used which both have k-best modes. Each program gives a list with its top k candidate moves (typically $k=3$ or something similar). Then the human has the final choice amongst the moves from the two lists of candidates.

In all three approaches the human does not only get the candidate moves, but also the corresponding principal lines and evaluations. Furthermore, the human is allowed to make the time management for the programs.

In 1996, Double-Fritz with Boss played 15 games and achieved a rating of 2520. Fritz ran on a PC with PentiumPro processor, at 200 MHz speed. The solo strength of Fritz on this machine was about 2350.

In 1997, List-3-Hirn with computer programs of strength 2530 in solo play had a match of eight games against Germany's No. 1 grandmaster Arthur Yusupov (Elo-rating 2640 in those days) and won by 5–3 [3]. This gave List-3-Hirn a provisional rating of above 2700.

For 1998 a match of List-3-Hirn against a world's top-10 human player was intended. However, after Yusupov's loss in 1997 top humans were no longer willing to compete with 3-Hirn.

On average, 3-Hirn and its variants were about 200 Elo-points stronger than the computers in the team (see Table 1). Our explanation for this success is that the strengths of human and computers were well combined in 3-Hirn.

The question marks "??" in the psychology column (Table 2) shall indicate that a human's awareness of the (human) opponent has advantages and disadvantages: he realizes when the opponent is nervous or angry or in time trouble. On the other hand, a tricky opponent may intentionally send false signals, or a "bold" opponent (like GM Kasparov is doing often) may make the human scared.

Table 1
Chronology of experiments with 3-Hirn

Year	Computer Elo-ratings	Number of games	Performance of 3-Hirn
1985	1500, 1500	20	1700
1987	1800, 1800	20	2050
1989	2090, 1950	8	2250
1992–1994	2260, 2230	40	2500
1995	2400, 2330	8	2550

Table 2
Strengths and weaknesses of programs, humans and 3-Hirn

Aspect/entity	Tactics	Memory	Long-range planning	Learning	Psychology
Program	Strong	Strong	Weak	Weak	??
Human	Weak	Weak	Strong	Strong	??
3-Hirn	Strong	Strong	Strong	Strong	Strong

In top-level correspondence Chess nowadays (in the year 2002) all players without a single exception use Chess programs for checking their analysis and improving playing strength. Such help is explicitly permitted (in the rules of the main German correspondence Chess federation "BDF" since summer 1996). For most players multiple-choice analysis is part of this computer assistance.

3. Conquering a new game with 3-Hirn

Starting with 3-Hirn in 1985, the author took more than 15 years to collect experiences with multiple-choice systems in chess. At the Dagstuhl workshop in February 2002 the introduction of a new game called "Clobber" gave the chance to try and study how multiple-choice systems may help to speed up the understanding of an entirely new game.

3.1. The new game "Clobber"

Clobber is a board game for two players White (o) and Black (x). Clobber is played on a rectangular grid of size $m \times n$. The players move in turn, with White to start. The last player to move is the winner. Draws are not possible.

In his turn a player has to jump orthogonally with one of his stones on a stone of the enemy which is placed on a directly neighbouring square. This piece of the enemy is deleted (it has been "clobbered"). In the starting position the board is completely filled with stones, alternatingly white and black ones. So, for instance on 6×5 board there are white stones on a1, a3, a5, b2, b4, c1, c3, c5, d2, d4, e1, e3, e5, f2, f4 and black stones on the remaining squares a2, a4, ..., f5.

In each move exactly one stone is clobbered. Hence, $m \times n - 1$ is a simple upper bound for the number of moves in a game starting with the full $m \times n$-board. Typically, a game in Clobber splits into several independent parts, and Combinatorial Game Theory may be applied to find game-theoretic values and optimal moves.

3.2. How Clobber was invented

In July 2001 R. Nowakowski (Halifax), M. Albert (Otago), and J.P. Grossman (MIT) analysed, a one-dimensional peg DUOtaire with single hops (so in contrast to the peg SOLItaire there are two players, moving alternatingly). Albert noticed that if the holes were paired up and at most one peg in each pair was allowed, one obtained a closed set of positions. Such positions can be represented with o = "o★" and x = "★o". So for example the peg arrangement "o★★o★o" becomes oxx . With this representation the legal moves are ox → .o and ox → x.

This inspired N & A & G to look at a two-player partisan game by having Left move x, Right move o, and allowing both players to clobber in either direction. The result was one-dimensional Clobber. Nowakowski, Albert, and Grossman became mainly interested in Clobber on a board with $2 \times n$ squares, with the starting position

```
xxxxxxxxxxx
ooooooooooo.
```

Computer runs for n up to 15 showed that the position is a first player win, whenever the length n is odd. A proof for general odd $n \geqslant 17$ still has to be found.

3.3. First computer programs for Clobber

In the Dagstuhl workshop J.P. Grossman organized a human Clobber tournament, whereas I decided ad hoc to "write" a computer program for playing the game. Not being an experienced programmer, my intention was to use "Zillions of Games" for this.

"Zillions of Games" is a programming environment for brain games. The user "only" has to specify the rules of a game in a "zrf" (a "Zillions Rules File"), in a Lisp-like manner. Then the Zillions engine generates a program playing this game more or less intelligently. This happens with the help of an automatically generated evaluation function and game tree search, including alpha–beta pruning and iterative deepening. However, Zillions does not understand the concept of combinatorial game theory with its "sums of games".

For all normal users the Zillions engine is a black box. Only the designers of Zillions, Mark Lefler and Jeff Mallett, know the inside structure. "Zillions of Games" runs under the Windows operating systems and has been commercially available since 1999 [11].

Clobber has very simple rules. It took me about 10 min to write a correctly working Zillions rules file. The graphical elements I simply took from the already existing

Zillions implementation of “Connect 4” (= “ Vertical_TicTacToe” in Zillions). Therefore, the white stones will be red in the graphic. To give an impression on the simplicity of Zillions, my complete code for 6×5 Clobber is included here.

```
***************   Clobber.zrf   ********************************
(game
  (title "Clobber 6x5 for Zillions  --  Dagstuhl Edition")

  (players White Black)
  (turn-order White Black)

  (board
      (image "images\Vertical_TicTacToe\VerticalTTT6x5.bmp")
      (grid
         (start-rectangle 1 1 30 30)
         (directions (n 0 -1) (e 1 0) (w -1 0) (s 0 1))
         (dimensions
              ("a/b/c/d/e/f" (30 0))
              ("5/4/3/2/1" (0 30))
         )
      )
  )

  (piece
      (name stone)
      (image White "images\Vertical_TicTacToe\red.bmp"
             Black "images\Vertical_TicTacToe\black.bmp")
      (moves
         (n (verify enemy?) add)
         (s (verify enemy?) add)
         (w (verify enemy?) add)
         (e (verify enemy?) add)
      )
  )

  (board-setup
     (White (stone a1 a3 a5 b2 b4 c1 c3 c5 d2 d4 e1 e3 e5 f2 f4))
     (Black (stone a2 a4 b1 b3 b5 c2 c4 d1 d3 d5 e2 e4 f1 f3 f5))
  )

  (loss-condition (White Black) stalemated)
)
*********************************************************
```

On a notebook with 233-MHz-Pentium II processor the Zillions program computes the game-theoretically correct outcome of a 6×5 Clobber game within a minute when already five or six pairs of moves have been made. In this early stage of a game a human typically has no clue about the game-theoretic value of the position.

Two other workshop participants wrote Clobber programs by their own: "Deep Clobber" by J.P. Grossman, and "Bobber" by R.A. Hearn (also at the MIT). "Deep Clobber" and "Bobber" were brute-force tree searching programs, with more or less no evaluation function except the knowledge about the win/loss status of end positions. They searched clearly faster than Zillions. A typical game between them and "Zilly" exhibited one of the following two patterns:

(i) At move pair 3 or 4 the brute-forcer found that the game was a win for him. Then the game was over.

(ii) At move pair 3 or 4 the brute-forcer found that the game was a loss for him. Then it played some move (for instance one which postponed the loss as much as possible). Now Zillions had to find winning moves for the next one or two or three times by its positional evaluation until it would also realize the final outcome.

A first tournament between the programs ended with a win of Deep Clobber. Two days later a second computer tournament was played, now on 6×6 boards, to get games with some more moves before knowing the final outcome. This time Bobber won, with Zillions being the runner-up.

3.4. 3-Hirn experiments in Clobber

The Zillions engine can be adjusted to play automatic games against itself. From watching several such Clobber games I got the feeling to understand some aspects of Clobber better than the program—although I had no chance in direct confrontation with Zilly. So I started thinking about the possibility of playing Clobber in 3-Hirn mode.

Two copies of the same deterministic programs will always give identical candidate moves. However, Zillions contains some random elements, and the user is allowed to adjust a parameter called "variability of play". I set variability on "large"—and in over 90% of the situations the two copies of Zilly proposed different moves. With these two copies I played games with "3-Hirn Good" versus "3-Hirn Bad". "3-Hirn Good" is the normal 3-Hirn: I tried to select the better one of the two candidate moves. To the contrary, in "3-Hirn Bad" I always selected the worse move. So, my overall intention was to produce games where "3-Hirn Good" won against "3-Hirn Bad". On 6×6 board four games were played, with "3-Hirn good" having White in two of them and the black stones in the other two. "3-Hirn good" won all four games. My human influence seemed to improve Zilly.

After the Dagstuhl conference I invested some afternoons to perform three more series of 3-Hirn experiments in Clobber. One advantage of Zillions is that multiple copies of the program may be run simultaneously on one computer without disturbing each other.

(A) "3-*Hirn Good*" *vs.* "3-*Hirn Bad*" *in* 6×5 *Clobber*: Hardware was a PC with 900 MHz Athlon processor. Each candidate move was based on a Zilly search with 5 s of computing time. Variability in Zillions was set on "large". I was the human with the final choice amongst the candidate moves. 2×6 games were played. The six games with “3-Hirn Good” for White gave a 5–1 win. The six games with “3-Hirn Good” on the black side gave a 4–2 win.

So, altogether “3-Hirn Good” beat “3-Hirn Bad” by 9–3. My control seemed to have the intended influence. Now I felt confident to perform matches with 3-Hirn against opponents who were not artificially weakened.

(B1) *3-Hirn vs. Zilly in* 6×5 *Clobber*: Hardware was a PC with 1000 MHz Athlon processor. In 3-Hirn each candidate move was based on a Zilly search with 5 s of computing time. Variability in Zillions was set on “large”. I was the human with the final choice amongst the candidate moves. The opponent, also Zilly, got 30 s per move, with variability also being set on “large”. As in series (A) 2×6 games were played. The six games where 3-Hirn was White resulted in a 2–4 loss. The six games with 3-Hirn on the black side gave a 5–1 win. So, altogether 3-Hirn (based on 5-s Zilly) beat Zilly (with 30 s) by 7–5.

For a comparison, a second experiment was conducted on the same hardware: Zilly with 5 s per move had to play against Zilly with 30 s per move. Variability in Zillions was set on “large” for both sides. The “short thinker” lost with 0–6 playing White, and with 2–4 playing Black. So altogether the 5 s version lost by 2–10.

(B2) *3-Hirn vs. Zilly in* 6×6 *Clobber*: Hardware was a PC with 900 MHz Athlon processor. In 3-Hirn each candidate move was based on a Zilly search with 5 s of computing time. Variability in Zillions was set on “large”. I was the human with the final choice amongst the candidate moves. The opponent Zilly got 15 s per move, with variability also being set on “large”. Again 2×6 games were played. The six games where 3-Hirn was White resulted in a 4–2 win. The six games with 3-Hirn on the black side also gave a 4–2 win. So, altogether 3-Hirn based on 5-s Zilly beat Zilly with 15 s per move by 8–4.

For comparison one more series of games was conducted on the same hardware: Zilly with 5 s per move had to play against Zilly with 15 s per move. Variability in Zillions was set on “large” for both sides. The “long thinker” won by 4–2 playing White, and got a 3–3 draw playing Black. So altogether the 5 s version lost by 5–7.

Hence, both in (B1) and in (B2) 3-Hirn on Zilly basis was clearly more successful than Zilly itself. $2 \times 2 \times 12 = 48$ single games do not provide much data. But, besides the raw results also the course of the games convinced me that 3-Hirn with Zilly and me is considerably stronger than Zilly alone. Let me close with the remark that as a lonely human I am rather chanceless against Zilly: running on a “slow” 233 MHz processor with 1 s thinking time per move Zilly beat me by a 6–2 margin on 6×5 board.

3.5. 3-Hirn strategies depending on the Clobber opponent

Near the end of a Clobber game the position tends to split into several independent groups. On larger boards the splits are typically nontrivial. Fig. 1 shows an example of a 14×14 board from a test game played by Zillions against itself. The position as a whole looks very complicated. However, it is split into 11 disjoint groups of small and medium sizes (the largest one—in the middle of the board—contains 24 stones). Combinatorial game theory allows to compute the value for each group and to “sum up these values” to obtain the win/loss-status of the whole position.

```
. . . . . . . x . . . . . .
. . . . o . . x o . . . . o
. . . x x . o . . . . x . o
. . . . . . o x x o x x . x
. . . . x o . . . x o . o o
. o x o . . . . . . . . o .
. . x . . . . . . . . . . .
. . o . x . o . x . x . o .
. . x x x x x x x x o . x x
. . . x x . . x x . . o . x
. x . x . . . . . . . x x .
. o . o . . . . . . . . . .
. . x . . . . . . . o o . .
. o o o . . . . . . . o x .
```

Fig. 1. A position on a 14×14 board with 11 disjoint groups.

Normal brute force programs and also Zillions have not incorporated the principles of Combinatorial Game Theory (CGT). In contrast humans can learn the concept of CGT rather easily. So on large boards with many disjoint groups humans should have an advantage over brute force tree searchers.

Open Question: which board size n^* is the turning point such that humans are superior to brute force programs on $m \times m$-boards with m larger than n^*? My personal experience is that an unshielded human has good chances of beating Zillions on the 12×12-board. Here my strategy is to split the game as soon as possible into small disjoint groups for which I know the CGT values.

A good 3-Hirn strategy (with two copies of Zilly on the computer part) for larger boards might depend on the nature of the opponent. Against a human (with CGT background) I would try to keep the position as connected as possible, or at least to preserve a giant connected component. In doing this my hope would be that the human was not able to compute the CGT value of the giant component, before Zilly had found a tactical win. Against a computer program without knowledge in CGT I would try to split the position into many (really) small pieces for which I knew the CGT values. Near the endgame I would try to select according to the "CGT-correct" moves.

3.6. New features for "Zillions of Games"

It was unpredictable and some sort of luck that the move candidates of Zillions in Clobber have enough variability to allow good 3-Hirn choices. In other games, even with zrf's of similar structure, variability of play may be much more limited.

The features below are a wish-list. Their realization might help to make Zillions an even more interesting tool for use in multiple-choice systems:

- Game tree search with a k-best mode would give k move candidates from a single engine.

- Alternative engines within Zillions would present move alternatives.
- The user should be allowed to tune certain parameters of the Zillions engine to obtain not only better but also alternative candidate moves.
- Zillions should compute and display principal lines. This additional information would allow a better understanding of the "intentions" behind the candidate moves.

Today, these points are already realized in many of the commercial chess programs.

4. Discussion

(1) Let us start with three theses.
- Combinatorial games are nice testbeds for decision support systems.
- Human (singular!) and computers (plural!) together may be much stronger than humans alone or computers alone.

  A problem with more than one human in a team is that typically humans are touchy when their work or their solutions are not accepted. In contrast, a computer program does not realize when you reject its proposal and make a reset or when you prefer other computer programs most of the time.
- In multiple-choice systems the human is not allowed to outvote the programs. This is an advantage.

  Of course this crude statement is exaggerated. Especially, it cannot be fully true when you look at current experiences in the game of Go [6,9]. What we mean is that almost all humans overestimate the importance of human influence in decision support systems. Especially it can really be an advantage for the human boss when he only has to select and does not have to look for even better alternatives. In [9] an overview is given on different grades of veto rights for multiple-choice systems.

(2) One of the most serious risks with multiple-choice systems is that of obtaining *micro mutations instead of true alternatives* [2]. Especially "simple" k-best algorithms tend to result in candidates which differ only in small details. The more fine-grained the set of legal moves the more serious this problem is. For instance, in Go "micro mutative candidates" occur more frequently on 19×19 boards than on those of dimension 9×9.

(3) Multiple-choice systems are used not only in Chess, Clobber, and Go. For instance, in the board game Othello (= Reversi) the top freeware program "W-Zebra" has a very comfortable multi-best mode. And for the solitaire card game FreeCell André Grosse and Stefan Schwarz have designed the freeware program "BigBlackCell" which gives in each situation an ordered list of all legal actions [10].

(4) Multiple-choice systems may be very helpful in many "serious" applications with strict real-time conditions. For example, in the fields
- optical pattern recognition
- vehicle routing and traffic control in general
- diagnoses in medicine, like for instance ECG interpretation
- language translation
- stock market actions and (financial) soundness checks.

But also in fields without severe real-time criticality, like

- recognition of protein structures and DNA alignments
- drug design
- theorem proving in mathematics

multiple-choice systems may help to improve the performance of "traditional" decision support systems.

Acknowledgements

Thanks are due to the chess grandmasters C. Lutz, G. Timoscenko, A. Yusupov, R. Knaak, and K. Müller for their willingness to participate in performance-oriented experiments with 3-Hirn and other multiple-choice systems.

My thanks also go to the Dagstuhl institution for initiating the fruitful seminar on algorithmic combinatorial game theory in February 2002, to J.P. Grossman and Robert Hearn for the thrilling competition in computer Clobber, and to Mark Lefler and Jeff Mallett for their concept of "Zillions of Games". Finally, J.P. Grossman and Georg Snatzke made helpful comments on earlier drafts of this paper.

References

[1] I. Althöfer, Das Dreihirn—Entscheidungsteilung im Schach. Computerschach & Spiele (December 1985) 20–22.

[2] I. Althöfer, On the *k*-best mode in computer chess—measuring the similarity of move proposals, ICCA J. 20 (1997) 152–165.

[3] I. Althöfer, List-3-Hirn vs. grandmaster Yusupov—a report on a very experimental match, ICCA J. 21 (1998) 52–60 (Part I) and 131–134 (Part II).

[4] I. Althöfer, 13 Jahre 3-Hirn—Meine Schach-Experimente mit Mensch-Maschinen-Kombinationen, published by the author, 1998, ISBN 3-00-003100-6.

[5] I. Althöfer, Decision support systems with multiple choice structure, in: I. Althöfer, et al., (Eds.), Numbers, Information, and Complexity, Kluwer, Dordrecht, 2000, pp. 525–540.

[6] I. Althöfer, Go mit dem 3-Hirn, Deutsche Go-Zeitung (July/August 2000) 40–42.

[7] I. Althöfer, Graded rights of veto in multiple choice systems, in: G. Johannson (Ed.), Proc. Eighth IFAC/IFIP/IFORS/IEA Symp. on Analysis, Design, and Evaluation of Human–Machine Systems, Kassel, 2001, pp. 657–663.

[8] I. Althöfer, C. Donninger, U. Lorenz, V. Rottmann, On timing, permanent brain, and human intervention, in: H.J. van den Herik, I.S. Herschberg, J.W.H.M. Uiterwijk (Eds.), Advances in Computer Chess 7, University of Limburg, Maastricht 1994, pp. 285–296.

[9] I. Althöfer, R.G. Snatzke, Playing games with multiple choice systems, Third Internat. Conf. on Computers and Games (CG'02), 2002, LNCS, submitted for publication.

[10] A. Grosse, S. Schwarz, Big-Black-Cell, downloadable at ⟨www.minet.uni-jena.de/∼BigBlackCell⟩. 2001.

[11] M. Lefler, J. Mallett, Zillions of Games, ⟨www.zillions-of-games.com⟩. Since 1999, weekly updated.