Multiple Choice Systems for Decision Support

Ingo Althöfer
Institue of Applied Mathematics
Faculty of Mathematics and Computer Science
Friedrich-Schiller-University Jena
07740 Jena -- Germany

1 Introduction

A human formulates a problem and starts one or several computer programs on it. The programs perform more or less complex computations and then present a clear handful of candidate solutions. From this candidate set the human makes the final choice. I call such a system *Decision Support System with Multiple Choice Structure* or shortly *Multiple Choice System*. In the game of chess Multiple Choice Systems have been very successful. The playing strengths were clearly above those of the "components" used in the Multiple Choice Systems. Based on these and other experiences, I believe that Multiple Choice Systems may be very successful in many fields of decision making.

2 Multiple Choice Systems in Chess

In a *3-Hirn* in chess one human and two different chess computers form a team. The human starts both machines and gives them an appropriate amount of time to present their candidate solutions. Then he selects one of these two proposals and realizes it. If both computers propose the same move (this happens in about 60 percent) this candidate has to be executed. So, the human controller is not allowed to overrule the machines. 3-Hirn plays chess against other humans or computers.
In *Double-Fritz with Boss* one human and only one chess computer are involved. However, this chess computer has a mode in which not only the best but also the second best move (according to its algorithm) are computed. The human makes the final choice amongst these two proposals. I called this construction Double-Fritz with Boss because the chess programm in my first such experiment had the name *Fritz 4*. In contrast to 3-Hirn the human almost always gets two different computer proposals. A problem is that these proposals are often somewhat similar to each other.
In *List-3-Hirn* the approaches of 3-Hirn and Double-Fritz with Boss are combined. Involved are one human and two different chess computers. Each program computes a list with, say, its three best move proposals. The human inspects both lists and selects a move for execution.

In 1985 I formulated the basic idea of 3-Hirn. Chess experiments took place from 1985 to 1997. In all these experiments I was the selecting human. The performance of the man-machines-teams was in average a whole class better than that of the chess computers used. And, at least from 1989 on, these chess computers were at least a whole class stronger than me. Highlights were a 5 : 3-match win of 3-Hirn over the International Master Dr. H. Reefschläger early in 1992, a 3.5 : 4.5-loss against Grandmaster C. Lutz in 1995, and a 4.5 : 3.5-win against Grandmaster G. Timoshchenko in autumn 1996 (using Double-Fritz with Boss). The final match was a 5 : 3-win of List-3-Hirn in "Shuffle Chess" over Germanys number 1 player, Grandmaster A. Yusupov in autumn 1997. Details may be found in my book "13 Jahre 3-Hirn" ("13 years with 3-Hirn"). Originally, for 1998 I had planned to play a match against a master in the top-10 of the world and, if successful, to challenge the world champion in 1999. However, after the match win against Yusupov I did not find an opponent.

The success of 3-Hirn and its variants in chess has at least two roots:

> (i) Humans think and make longterm plans. At the same time we are susceptible to "unforced errors". Computers are technically precise, but they are unable to plan. When a human selects from computer proposals unforced errors are (almost) excluded, and the human may try to realize his longterm plans. Hence, the strengths of human and computer are combined.
>
> (ii) A human takes into account against whom he is playing. Against other humans the controller of 3-Hirn may aim at positions which are complicated and in which unforced errors are likely to happen. In the matches with humans I often selected moves according to the maxim of former vice world champion David Bronstein: "Don't solve problems, but create them." In play against computers the controller of 3-Hirn should try to reach quiet positions in which longrange planning is the dominating factor.

Commercial chess computers are available since 1977. In the mid 90's chess programs came up which were designed not mainly for autonomous play but as tools for human analyses of chess positions. The starting point was in 1993 when the program "Doctor?" was the first to have a k-best mode. In this mode not only the best but the k best moves (according to the opinion of the program) are computed. A preliminary climax of this development is the program "ChessBase 7.0" which allows to run several chess programs simultaneously, each one in some k-best mode.

3 Examples of Multiple Choice Systems in Other Fields

In a few other fields Multiple Choice Systems are established. In this section five examples are listed, all dealing with traffic and transportation.

> (a1) The software company AND distributes a program "FLIGHT WORLD" which proposes flight connections for arbitrary destinations all over the world. If there are alternative connections they all are shown, ordered by increasing total connection time.
>
> (a2) Not every town has an airport. For this case the AND program has an option to show a list of near airports, ordered by reachability. Leipzig, Erfurt, Berlin-Tegel, Berlin-Tempelhof, Rostock-Laage, and Berlin-Schönefeld are, in this order, the entries for Magdeburg.
>
> (b) Since the mid 90's, there are routing programs for road traffic. The user enters starting point, destination, and perhaps some more parameters (criteria: "shortest" route or "fastest", average speeds for certain types of roads, ...). Then the program makes a routing proposal. Many routing programs by several companies are competing in the market. Only the products of AND allow to compute up to three alternative routes. For Jena → Magdeburg "AND Route World" proposes the following routes:

(i) "The eastern highway":	Lobeda, A4, A9, A14, B71;	193.6 km;	2:37 h:min.
(ii) "The north-western route":	B88, B180, B81;	158.2 km;	2:44 h:min.
(iii) "Eastern highway with shortcut":	B7, A9, A14, B71;	180.0 km;	2:37 h:min.

> By the way, in this program it happens rather often that best and third-best proposal are more similar to each other than to the second-best. The program seems to have order to make especially the second-best route clearly different from the best one.
>
> (c1) The German railway company "Deutsche Bahn" regularily presents its time table on CD-ROM. For given starting point, destination and time window the underlying program computes a list of good train connections.

(c2) A practical auxiliary tool on this railway CD (and also a nice toy) is the phonetic search for railway stations. When you enter for instance *Madburg*, the following ordered list of candidate stations is presented on the screen: 1. Bedburg-Hau, 2. HAMBURG, 3. Bedburg (Erft), 4. Hildburghausen, 5. Maulburg, 6. Hamburg Hbf, 7. Magdeburg Hbf, 8. Marburg Süd, 9. Moosburg, 10. Duisburg Hbf, 11. Augsburg Hbf, and so on.

4 Visualisation in Multiple Choice Systems

In Multiple Choice Systems a good visualisation is much more important than in normal programs as the controlling human has to inspect not only one but several candidate solutions simultaneously. For instance, in case of three proposals it is not appropriate to present each of them in the same format as a single solution would be done. Especially, relevant differences between the candidates should be marked or emphasized to support the process of choice. In real time systems a good visualisation is extremely neccessary to avoid that the human controller becomes a brake-block.

5 Computing K Alternatives

Given an optimization task, a "normal" algorithm computes *the best* solution. If alternative solutions are wanted in a discrete problem, it is natural to search not for the best but for the *k best solutions*. K-best algorithms have been designed and studied since the late 50's. The internet bibliography of D. Eppstein gives a good survey.
"Normal" K-best algorithms suffer from one problem when it comes to applications: The k solutions are often very similar to each other and not true alternatives (an example for such micro mutations: 1. Red Ferrari 2. Red Ferrari with ash-tray). In papers with W. Wenzel *k-best-optimization under distance constraints* was introduced and studied exemplarily for matroids and their generalisations: A distance function is defined on the set of all feasible solutions, and only those k-tuples are admitted for which the k solutions have pairwise distances greater or equal to a parameter d* which is given by the user. The larger d*, the more different the k solutions have to be.
Maybe, "*k-alternative algorithms*" is a good sum-up name for all algorithms which do not simply compute the k best solutions but k good and interesting and clearly different alternatives.

6 On the Right of Overruling Computers

Often people have asked why I did not allow myself to overrule the computers in the chess 3-Hirn. Indeed, it took me some time to accept the role of only selecting from computer candidate solutions. But the missing right to overrule even became a relief when I no longer looked for move proposals of my own and instead relied on those given by the machines. I could fully concentrate on long range planning. However, I can imagine that people with a stronger EGO have problems to accept such a "rather passive" role.
The German railway company "Deutsche Bahn" is currently introducing new structures of computer-based control. Part of the software is a program which proposes decisions to the managing director in a railway station, for instance "let train Y not wait for the late train X". The director is allowed to overrule the program. But in this case a documentation window opens on his computer screen and he has to write a (short) explanation for his deviating decision.

7 Short Outlook

Naively, one might try to use *Multiple Choice Systems* in every situation. But this can only work, if appropriate software is available (independent programs or programs with a good k-alternative mode). Furthermore, human experts may be so strong that a pure multiple choice mode would not allow them to fully bring in their expert's knowledge. A good example is computer-aided chess when instead of amateurs human Grandmasters are sitting on both sides of the board (Friedel 1998), (Althöfer 1999). Grandmasters use the computers in much more sophisticated ways than I did it as a hobby player.
Currently it happens more or less by chance when Multiple Choice Systems are used in a specific field for decision support. One future direction of research will be to check systematically under what circumstances *k-alternative algorithms* and *Multiple Choice Systems* are useful tools.

8 Rules of Thumb

* Quality of decision and creativity in "Multiple Choice Mode" are possible.

* It takes time to learn selecting between alternatives.

* Presentation and visualisation of alternatives is a very important aspect in Multiple Choice Systems.

* Often the k absolutely best solutions are not optimal sets of candidates. *Alternatives* should be more than only micro mutations of each other.

* Of course there are fields where Multiple Choice Systems are not the best approach.

Literature and Software

I. Althöfer. 13 Jahre 3-Hirn: Meine Schach-Experimente mit Mensch-Maschinen-Kombinationen. Published by the author, Jena 1998, ISBN 3-00-003100-6.

I. Althöfer. Advanced shuffle chess with technical improvements. Journal of the International Ccomputer Chess Association, Vol 22.4 (December 1999), pp. 245-251.

I. Althöfer. Decision Support Systems with Multiple Choice Structure. In "Numbers, Information, and Complexity" (Editors I. Althöfer, N. Cai, and others). Kluwer, Amsterdam, 2000, pp. 525-540.

I. Althöfer and W. Wenzel. 2-best solutions under distance constraints – the model and exemplary results for matroids. Advances in Applied Mathematics 22 (1999), 155-185.

AND Software GmbH. Car routing program *Route World*. Wiesbaden, 1998.

AND Software GmbH. Flight routing program *Flight World*. Wiesbaden, 1999.

ChessBase GmbH. Computer chess program *ChessBase 7.0*. Hamburg, 1998.

Deutsche Bahn. *Elektronischer Fahrplan Deutschland und Europa*. Frankfurt a. M., 1999.

D. Eppstein. Internet bibliography on k-best algorithms. Size: 398 kilobytes at April 5, 2000. http://www.ics.uci.edu/~eppstein/bibs/kpath.bib .

F. Friedel. Advanced chess by Kasparov and Topalov. ICCA Journal, 21.2 (June 1998), 126-130.