REPRESENTATION OF EXPERTS' KNOWLEDGE IN A SUBDOMAIN OF CHESS INTELLIGENCE

H.J. van den Herik
Delft University of Technology
Department of Mathematics and Informatics
Julianalaan 132
2628 BL DELFT / THE NETHERLANDS

0. ABSTRACT

This paper considers how to represent rule knowledge in a chess-endgame procedure. The procedure, to be used jointly with depth-first searching, deals with the King, Bishop and Knight versus King (KBNK) endgame; the rules derive from theory books, discussions with (grand)masters and the author's experience.
The knowledge about rules is structured: partitioning into patterned equivalence classes (Bramer, 1975, 1977) is extended so as to make the procedure act similarly to a human expert, who, in the KBNK domain, prefers to be guided by patterns rather than by exploring move sequences in depth.
The procedure, applied by incorporation in PION, a chess-playing program, has been tested and commented upon by Averbach and others and has proved to execute correct mates in all cases submitted. It is concluded that, within its domain, its level is that of a chess expert though no proof is available that its play is optimal.
KBNK has been solved constructively by a supplementary data-base approach, establishing it as a 33-move game in the maximin sense. This result, as well as some others presented here, are new to the theory of chess. The research reported may well lead to improving experts' play.

1. INTRODUCTION

While the theory of chess is continually being added to, it is, by the same token, far from perfect. In the last few years, a new element has entered: extension of the theory no longer depends on human thoughts and ideas *only*. In the chess field, AI-programs and problem-solving programs, consulting a pre-constructed data base, have begun to contribute. The strategy is known to depend on the concepts the player has in mind. For example, a grandmaster "knows" how to handle a certain position and this is why he immediately "sees" the "right" move (De Groot, 1965). Extending the body of knowledge capable of supporting the chess-player's concepts may result in an expert's playing:

(i) *acceptably* in positions where no guidance was available before;
(ii) *correctly* in positions which were not sufficiently explored before;
(iii) *optimally* in positions exhaustively charted by the knowledge supplied.

The fundamental question reads as follows: is it possible to achieve similar results by extending the knowledge of a chess program as by extending the knowledge available to human experts? For some types of endgame, the answer is in the affirmative, though the process of extending a program's knowledge is far from trivial (Bramer, 1977; Bratko & Niblett, 1979; Van den Herik, 1980 b).

The knowledge to be supplied is extensive and highly detailed, e.g., the rules in the KBNK endgame need to be very explicit even if the King Alone is pinioned since he may be in the *wrong* corner). This paper will report the results of a new method (Van den Herik, 1980 b, 1982) of representing knowledge, combined with conventional depth-first searching, to the special domain of the KBNK endgame.

2. THE STRUCTURE OF THE ALGORITHM

Bramer (1975), introducing the concept of *equivalence classes*, did not admit any searching beyond a depth of *one* ply. In his view, in an elementary endgame, such as King and Pawn vs. King (KPK), the program should rely on explicit knowledge only. Even for KPK, Bramer's program needs 20 classes for a correct strategy and 38 for an optimal strategy.
As the complexity of the endgame increases, a point is soon reached where the number of equivalence classes grows beyond reason (Van den Herik, 1980 a). In order to keep the number of classes (embodying knowledge) down to manageable proportions for more complex endgames, the concomitant use of deeper searching is appropriate. This combination has proved succesful in a number of four-piece endgames, such as KNPK, KBNK and KBPK.
Our basic move-finding algorithm (in which White is assumed to be the stronger side and to move first), derived from Bramer's algorithm, is structured as follows:

a) generate the set Q of all immediate successor positions of a position p;
b) find the highest ranked element of Q, say q;
c) play the move corresponding to q.

In step b), *find* means 'determine by goal-directed search'. In accordance with Bramer, we induce a ranking on Q by defining an overall ranking on the set Q^*, being the set of all legal BTM (Black to move) positions. We assign each position in Q to exactly one of a number of disjoint subsets, exhaustively partitioning Q^*. The aim is for each subset of the *partition*, termed a *class*, to correspond to some significant recognizable *feature* of the endgame as perceived by chess-players, e.g., 'Black is in check'. The connection between 'partition', 'feature' and 'class' is given by regarding features as equivalence relations, on which a fundamental theorem reads: "Let F be an equivalence relation on Q^*. Then the quotient set Q^*/F is a partition of Q^*." This allows the programmer to list distinct static board·features F_1, F_2, F_3, ..., etc., which induce a partition in the above sense. Therefore, for all q:

$$F_i \supset \neg F_j \text{ for all } j \neq i.$$

Using <u>if</u>-<u>then</u> rules, the construction is as follows:

```
if F1 then posvalue := classvalue[1]
else
if F2 then posvalue := classvalue[2]
else
if .....
```

Notice that in this construction the guaranteed order of the execution of the tests allows us to simplify the descriptions of the composite predicate functions F_i, because it is now true that

$$F_i \supset \neg F_j \text{ for all } j < i.$$

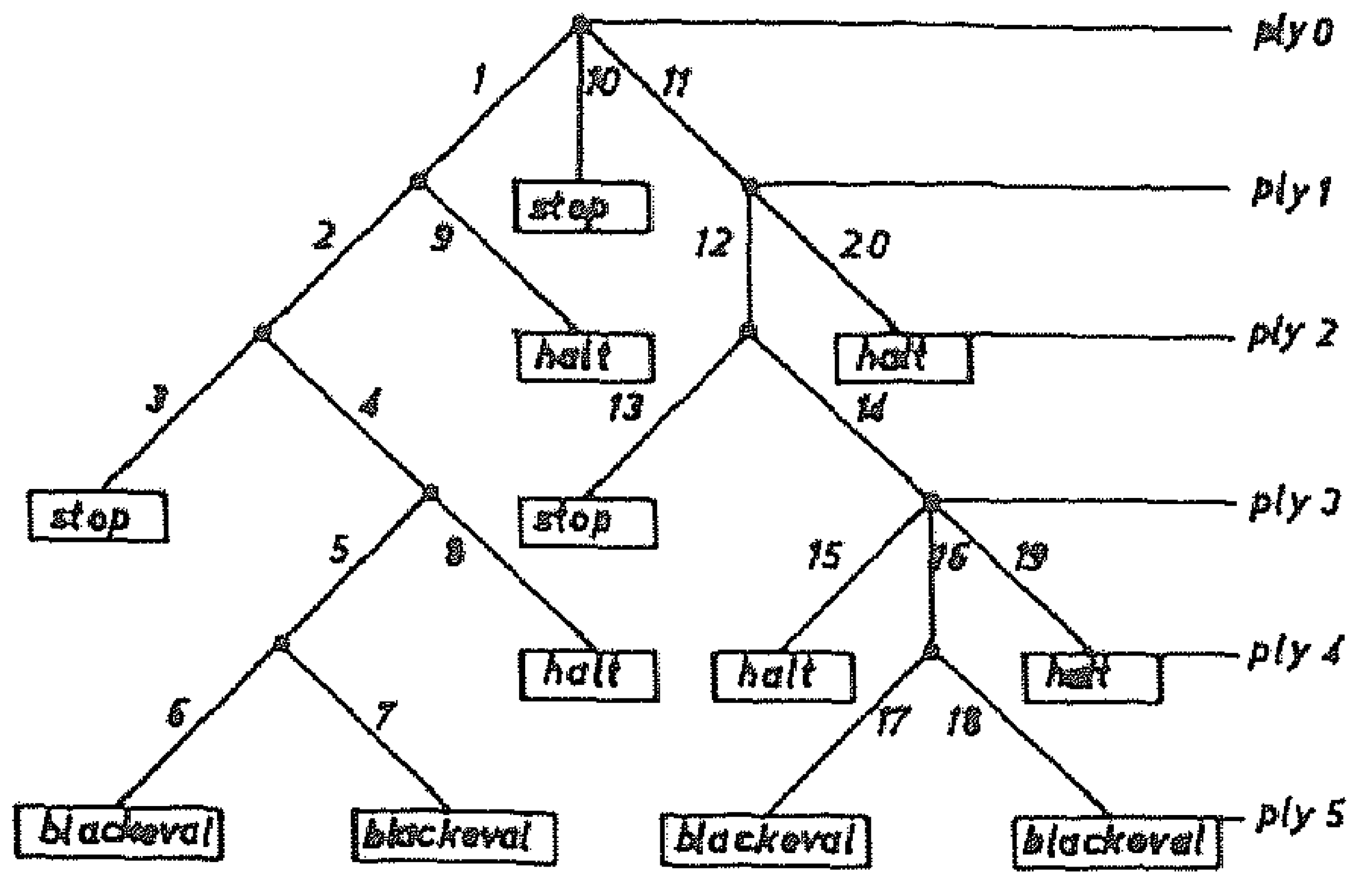
In order to effect the closure of the equivalence relation on Q^*, one introduces a residual class with the empty feature F_n such that

$$F_n \equiv (\neg F_j \text{ for all } j < n).$$

If a position belongs to the residual class, the search must be continued; no assignment should be made at this point, an analogous procedure starts with Black to move and Black's set of F_i. If two different moves lead to positions which belong to the same class (other than the residual class), a specific scoring function must be applied in order to break ties. If the tie persists, the first position examined will be chosen. Of course, a maximum depth of searching has to be decided in advance. Having reached the maximum depth values must be assigned to each position so reached. This means that we need two more sets of equivalence classes for *final* positions with White and Black to move respectively.
The four sets of equivalence classes required to construct an evaluated search tree may be whittled down to three when one determines that a specific side is always to move in the maximum depth position. We introduce the following names for our sets.

stop : the set of patterns (features of positions) with Black to move applicable when the value of the position investigated can be determined straightforwardly;
halt : similar to *stop*, but with White to move;
blackeval: the set of patterns which differentiate among the positions with Black to move when the maximum depth of searching has been reached and the positions must be evaluated;
whiteval : similar to *blackeval*, but with White to move.

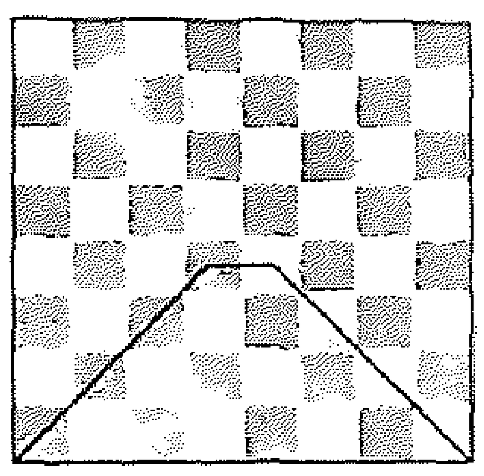
The introduction of these four sets of evaluation functions leads naturally to a tree structure of which the following is a typical instance.
[Our example is a 5-ply tree; the numbering of the branches reflects the sequence of traversal in this instance.]



In order to prevent the generation of inappropriate moves, a heuristic function, called *reject*, has been introduced which trims the generator of all legal moves into a generator of all plausible moves for the endgame in question. In summary, the algorithm constructs a tree with a variable but limited depth. The fanout varies with the level (ply). This and the actual depth depend on the given position.

3. THE KBNK PROGRAM

In a previous experiment, we used the KNPK ending with the Pawn on the h-file as a test-bed for the representation of chess-endgame knowledge (Van den Herik, 1980 b, 1982). One of the possible generalizations mentioned in the 1982 article was the application of that technique to the KBNK endgame. The resulting KBNK program, HEDEAM, has been designed and implemented by Van den Herik, Dekker and Ampt. It is based on the theory for this specific endgame as pioneered by Delétang (1923) and very systematically described by Chéron (1964). Dekker and Ampt (1981) have recast this theory for specific use in an endgame program.



The assumptions in our model are
(i) White possesses the Bishop and Knight;
(ii) White's Bishop is on a black square;
(iii) The black King (BK) is within the trapezium a1-h1-e4-d4.

DIAGRAM 1.

None of these assumptions detracts from generality: (i) is circumvented by relabelling sides; (ii) can be complied with by reflection in the vertical midline, while (iii) can always be made to obtain by mirror reflection in one of the diagonals or both.

In *stop*, *halt*, *blackeval*, *whiteval* and *reject*, chess knowledge from theory books, grandmasters and the author's experience is accumulated. The numbers of patterns in the evaluation functions total: 17 in *stop*, 3 in *halt*, 4 in *blackeval* and 4 in *whiteval*. A full description of the patterns in these functions is given by Van den Herik (1983).

In order to convey some idea about the partitions by class, we cite the first part of *stop*, with the patterns transposed into human terms.

class 1: Bishop or Knight are no longer on the board
class 2: Mate

class 3: Stalemate
class 4: Bishop or Knight are under attack
class 5: BK locked in the small Bishop triangle
class 6: BK locked in the proper corner with a
Bishop at a distance
class 7: BK locked in the middle Bishop triangle
⋮
class 11: BK locked in the large Bishop triangle
and so on.

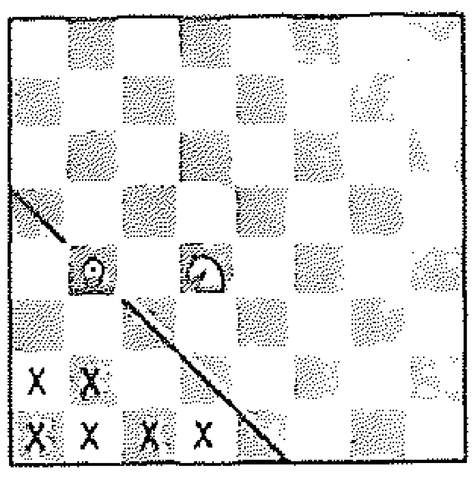


DIAGRAM 2
The middle Bishop triangle

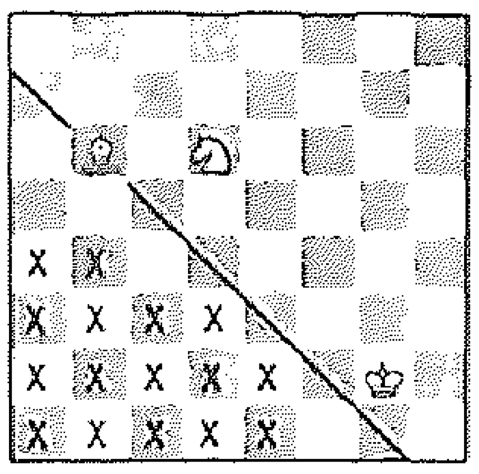


DIAGRAM 3
The large Bishop triangle

[x indicates possible positions of the Black King]

The flavor of *stop* is possibly illustrated by a minor extract: if none of the preceeding classes apply, membership of class 5 is the truth of

```
((bk<c1)&&(wb==a3 ||wb==c1)&&(wk!=refl3sq(wb)
&&(d(wk,wb)<3)&&(d(wk,c2)<=d(bk,c2) || nd(c2)==1)
&&(d(wk,b3)<=d(bk,b3) || nd(b3)==1));
```

for three lines out of 39, and incidentally involving refl3sq(wb) for reflection (iii) for generality.

4. HEDEAM's PERFORMANCE

The performance of HEDEAM hasbeen tested by several masters and grandmasters, Averbach, Elo, Enklaar and Sosonko among them. It has passed the expert test many times, occasionnally *summa cum laude*, producing a more direct way to mate than was foreseen by a (grand)master. Evenso, its play, while correct, falls short of the optimal, as proved by the technique of 5, below. Yet, International Grandmaster Yuri Averbach, one of the world's best endgame specialists, was impressed by HEDEAM's systematic approach to forcing a mate, for example for the following position (Dekker & Van den Herik, 1982 a).

White: HEDEAM (Ka8 Bh1 Nh8)
Black: Averbach (Kd4)
TH Delft, 19 mei 1982
1. Kb7 Kc5 2. Nf7 Kb5 3. Ne5 Kc5 4. Kc7 Kd4 5. Kd6 Kc3 6. Kc5 Kb3 7. Kd4 Kb2 8. Kd3 Kc1 9. Nc6 Kb2 10. Nd4 Kc1 11. Kc3 Kb1 12. Nb3 Ka2 13. Be4 Ka3 14. Bb1 Ka4 15. Nd4 Ka3 16. Nb5+ Ka4 17. Kc4 Ka5 18. Bf5 Ka4 19. Bd7 Ka5 20. Nc7 Kb6 21. Nd5 Ka5 22. Kb3 Ka6 23. Ka4 Kb7 24. Ka5 Ka7 25. Bc8 Kb8 26. Ba6 Ka7 27. Ne7 Ka8 28. Kb6 Kb8 29. Nc6+ Ka8 30. Bb7 mate

The applicability of HEDEAM is ensured by its being incorporated as an endgame procedure in the chess program PION (TH Delft). The interfaces are described in Derksen and Huisman (1982).

5. KBNK IS A 33-MOVE ENDGAME

The KBNK endgame can be regarded as a constructive maximin problem: what is the maximum of the minimal number of moves *necessary and sufficient* for White to mate the black King Alone, starting from an arbitrary position with White to move and assuming optimal counterplay by Black?
The estimates of chess publicists diverge considerably: Pachman conjectured 32, Rabinovich 33, Euwe/Donner 34, Fine 34, Golombek 'about 34', Averbach '*etwa* 35', Znosko-Borovsky 40. No specification is found in Bijl, Chéron, Keres and Withuis. Computer chess researcher Bramer (1982) stated: '... the endgame King, Bishop and Knight versus King is thought to require up to 34 moves to win and an error in certain critical positions can easily lead to an exceeding of the 50-move limit.'

Prompted by the prevailing uncertainty, Dekker and Van den Herik decided to build a data base for the KBNK endgame solving the maximin problem, in fact by exhaustive enumeration. Their findings have been published in 'Computerschaak' (Dekker & Van den Herik, 1982 b, 1982 c) stressing the chess-theoretical aspects and in the HCC Nieuwsbrief, (Dekker & Van den Herik, 1983 a, 1983 b) with stress on the computer-science aspects.

The most important results are
1. The KBNK endgame has a 33-move maximin solution.
2. Rabinovich was the only one to have this predicted.
3. No chess publicist (not even Rabinovich) had published an analysis optimal in the maximin sense.
4. There are 1104 'mate in 33' positions.
5. Not a single 'mate in 33' position had been published before, let alone a 'mate in 33' solution.
6. Black's best counterplay often involves his taking refuge in a corner the extreme square of which is opposite in colour to White's Bishop's.
7. Forcing the King from the large Bishop triangle to the middle Bishop triangle (the method of Delétang) need not represent an optimal method of play.
8. The KBNK data base contributes to the theory of chess by providing new patterns.

6. FUTURE RESEARCH

The implementation of the KBNK endgame, as embodied in HEDEAM, has also led to an improvement of endgame-program construction by sparking a new idea in the compilation of knowledge to be fed to programs. In order to be able to deal with a large amount of endgame programs, we are working on a pattern compiler pursuing the following aims:
a) to make the pattern files readable to a chess-player who is not a computer specialist;
b) to give chess-players of this kind the opportunity to construct, in person, a pattern file for a given endgame;
c) to be able to implement all subsequent programs much faster.

A fuller description of this idea has been published by Van den Herik (1983).

Another research item is based on a contrast between pattern-driven HEDEAM and the brute-force data-base approach. Whenever HEDEAM deviates from the optimal path, it is intended that the program's rules should be refined with the ultimate aim of refining the program's strategy, now correct, into an optimal one in Bramer's 1982 sense of these terms.

7. ACKNOWLEDGEMENTS

The author wishes to thank Prof.dr. I.S. Herschberg for his many valuable suggestions on the substance of the research described in this paper and its predecessors. He is also grateful for Herschberg's careful scrutiny of the English text. The co-operation with the students Sito Dekker and Gilles Ampt has inspired all three of us; it was Sito Dekker, in particular whose initiative led to many of the results in this paper. The competent technical assistance of Jan Roerade and of Ms. J.W. Pesch who did all the typing is also gladly acknowledged as having been essential to the production of this publication.

8. REFERENCES

BRAMER, M.A. (1975). Representation of Knowledge for Chess Endgames. Technical Report, The Open University: Faculty of Mathematics, Milton Keynes, England.

BRAMER, M.A. (1977). Representation of Knowledge for Chess Endgames: Towards a Self-improving System. Ph. D. Thesis, The Open University: Faculty of Mathematics, Milton Keynes, England.

BRAMER, M.A. (1982). Refinement of Correct Strategies for the Endgame in Chess. SIGART Newsletter, Special issue, Game-playing Programs: Theory and Practice (ed. M.A. Bramer), pp. 155-163.

BRATKO, I. & NIBLETT, T. (1979). Conjectures and refutations in a framework for chess endgame knowledge. Expert Systems in the micro electronic age (ed. D. Michie), pp. 83-101. Edinburgh University Press, Edinburgh.

CHÉRON, A. (1964). Lehr- und Handbuch der Endspiele, Band II. Zweite verbesserte Auflage. Printed in Germany (1973), pp. 10-13. Siegfried Engelhardt Verlag, Berlin 28, (Frohnau).

DEKKER, S.T. & AMPT, G.J. (1981). Het matzetten met loper en paard. Computerschaak, Jrg. 1, No. 5, pp. 32-37.

DEKKER, S.T. & HERIK, H.J. VAN DEN (1982 a). Averbach en Elo in Delft. Computerschaak, Jrg. 2, No. 4, pp. 122-124.

DEKKER, S.T. & HERIK, H.J. VAN DEN (1982 b). Mat met loper en paard. Computerschaak, Jrg. 2, No. 4, pp. 129-138.

DEKKER, S.T. & HERIK, H.J. VAN DEN (1982 c). Toch mat in 33. Computerschaak, Jrg. 2, No. 5, pp. 167-168.

DEKKER, S.T. & HERIK, H.J. VAN DEN (1983 a). De konstruktie van een database voor het matzetten met loper en paard (1). HCC Nieuwsbrief 46, Jrg. 5, No. 12, pp. 53-56.

DEKKER, S.T. & HERIK, H.J. VAN DEN (1983 b). De konstruktie van een database voor het matzetten met loper en paard (2). HCC Nieuwsbrief, in press.

DELÉTANG, D. (1923). Mat avec le Fou et le Cavalier. La Stratégie, Vol. 56, No. 2, pp. 25-32.

DERKSEN, J. & HUISMAN, J. (1982). Het schaakprogramma PION. Vorderingen in 1981/1982. Onderafdeling der Wiskunde & Informatica, TH Delft, Delft.

GROOT, A.D. DE (1965). Thought and Choice in Chess (ed. G.W. Baylor). Second edition 1978. Mouton Publishers, The Hague-Paris-New York.

HERIK, H.J. VAN DEN (1980 a). Computerschaak, Direkte evaluatie door middel van patroonherkenning in het KNPK-eindspel. Technische Hogeschool Delft, Delft.

HERIK, H.J. VAN DEN (1980 b). Goal-directed Search in Chess End Games. Delft Progress Report, Vol. 5, No. 4, pp. 253-279.

HERIK, H.J. VAN DEN (1982). Strategy in Chess Endgames. SIGART Newsletter, Special issue, Game-Playing Programs: Theory and Practice (ed. M.A. Bramer), pp. 145-154.

HERIK, H.J. VAN DEN (1983). Computerschaak, Schaakwereld en Kunstmatige Intelligentie. Ph.D. Thesis, Technische Hogeschool Delft. Academic Service, 's-Gravenhage.