



# THE UNIVERSITY *of* EDINBURGH

This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e.g. PhD, MPhil, DClinPsychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

HUMAN AND MACHINE  
REPRESENTATIONS OF KNOWLEDGE

by

Danny Kopec

Ph.D.

University of Edinburgh

1982



## Abstract

Four existing knowledge-representations for the computation of similar functions in a chess endgame were implemented on the same computer in the same language. They are compared with respect to efficiency regarding time-space requirements.

Three of these programs were then paraphrased into English and all four were studied for their feasibility as 'open book' advice texts for the human beginner in chess. A formally verified set of rules was also tested for its suitability as an advice text. The possible effectiveness of these advice texts in 'closed book' form is considered.

The above experiments comprise a case study of a phenomenon known as the "human window". This phenomenon motivated an analysis of four documented instances of mismatch between human and machine representations. These are:

- I Three Mile Island
- II Air Traffic Control
- III NORAD Military Computer System
- IV The Hoogoven Royal Dutch Steel automation failure

### ACKNOWLEDGEMENTS

I am indebted to Professor Donald Michie for his persistent guidance, and encouragement. There are also many people without whose assistance I could not have progressed. Foremost were my friends and colleagues Alen Shapiro, Timothy Niblett, and Leon Sterling; Arthur Biagi and the students at Holyrood H.S., Mr. McDougall, Assistant Head Teacher at James Gillespie's H.S. and his students, Martin Feather, and Michael Clarke at Queen Mary College. Last but certainly not least of all I thank Jean Duckman for painstakingly preparing most of the typescript.

I express appreciation to all my colleagues over the years at the Machine Intelligence Research Unit and Edinburgh University for their moral support and help with facilities. The work reported in Chapter IX was made possible by a 13-month grant from the Commission of the European Communities, Sub-programme FAST, for which I am most grateful.

Dedicated to my parents, Vladimir and Magdalena Kopec,  
who have supported me from the beginning,  
and to the memory of Dr. Howard Cohen and those  
who weren't as lucky as me.

# TABLE OF CONTENTS

Page No. (s)

INTRODUCTION	1
I BACKGROUND	11
1.1 KPK From the Chess Master's Point of View	11
1.2 History of KPK Programs	13
1.3 Huberman's Work	15
II INADEQUACY OF THE STANDARD ALGORITHMIC APPROACH TO COMPLEX PROBLEM-SOLVING	21
2.1 Introductory Remarks	21
2.2 Harris' KPK Program	22
2.1.1 Verification and Changes	22
2.1.2 Tests on 1001 Configurations	24
2.1.3 Tests on Random Samples	25
2.1.4 Improving Efficiency	30
2.1.5 Vital Statistics	33
2.1.6 Conclusions	36
III DESCRIPTION OF THREE NON-ALGORITHMIC APPROACHES TO KPK	37
3.1 Beal's KPK Program	37
3.2 Bramer's KPK Programs	40
3.3.1 Goals	40
3.3.2 The Model	41
3.3.3 Implementation of the Model for KPK	43
3.3 Clarke's KPK Database	47
3.4.1 Construction and Organization	47
IV COMPARING THE FOUR REPRESENTATIONS ON COMPUTATIONAL EFFICIENCY	51
4.1 Their Implementation on the DEC-10	51
4.2 Spatial Factors	53
4.3 Time Factors	54
V EARLY EXPERIMENTS	56
5.1 Objectives	56
5.2 Design of the Experiment	58
5.3 Method: The Translation Process	60
5.3.1 Translation of Beal's Program	60
5.3.2 Examples of Beal's Decision Table Approach for KPK	63
5.3.3 Translation of Harris' Program	64
5.3.4 Translation of Bramer's Program	65
5.3.5 Example of Bramer's 19 Equivalence Approach for KPK	66
5.4 The Advice Texts	68
5.4.1 The Beal Advice Text	69
5.4.2 The Harris-Kopec Advice Text	78
5.4.3 The Bramer Advice Text	91
5.4.4 Bramer's KPK Program as a Human Window Example.	100
5.5 Results of Early Experiments	103

5.6 Summary and Conclusions	110
VI THE CLARKE ADVICE TEXT	114
6.1 Objectives	114
6.2 Experimental Design	116
6.2.1 Example of position retrieval from the Clarke Database	119
6.3 Results	120
6.4 Conclusions and Discussion	125
VII THE NIBLETT ADVICE TEXT	133
7.1 Objectives	133
7.2 Design and Procedure of the "Open Book" Experiment	134
7.3 Results of the Open Book Experiment	142
7.4 Design and Procedure of the "Closed Book" Experiment	145
7.5 Results of the Closed Book Experiment	146
7.6 Conclusions and Discussion	148
VIII OVERALL SUMMARY OF CONCLUSIONS	151
IX CASE STUDIES FROM REAL WORLD APPLICATIONS	153
9.1 Three Mile Island	156
9.2 Air Traffic Control	158
9.3 NORAD Military Computer	165
9.4 Royal Dutch Steel	168
9.5 Overview	172
REFERENCES	174
Appendix A.1 : Correction of Type 0, Type 1, and Type 2 Harris Program errors	
A.2 : Correction of Type 3 and Type 0.n Harris Program errors.	
Appendix B : The KPK MANUAL	
Appendix C : Performance of 8 rated control subjects on a KPK Quiz.	
Appendix D : Pre-Test and Post-Test performance by a small sample of rated human chessplayers.	
Appendix E : Notes	

## INTRODUCTION

This thesis concerns itself with the comparison of knowledge representations for a specialized sub-domain of chess, namely the ending King and Pawn versus King (KPK). The side with the pawn will be referred to throughout as White. The game of chess has not been solved, even with the aid of powerful computing methods, and may possibly not be feasibly soluble in the sense of, for example, Knuth (1976). The total problem space of KPK contains only 98304 legal piece configurations after allowing for right-left symmetry, not distinguishing "White-to-move" from "Black-to-move" positions, and not including positions where the pawn is on the eighth rank with White-to-move (Clarke, 1977). It may seem unclear how study of so small a fragment of so large a problem can significantly contribute to the computational study of chess as a whole.

It is generally accepted that the endgame is the phase of chess where classes of players are the most clearly separated, i.e. the hardest and most subtle phase. Therefore this is also the phase where domain-specific knowledge is most significant. As a miniature illustration of this, how is it that masters can recognize the value and correct move(s) in almost any KPK position at a glance? Certainly masters have not studied or memorized all 98304 legal piece-configurations. Nor is it feasible for a master in the general case to perform the recognition by ex-



haustive lookahead calculation: in worst case the number of nodes on the requisite search tree greatly exceeds  $10^{14}$ . Instead there must be some pattern store or "reference library" which masters can consult internally when considering any KPK position. While it is true that calculation as opposed to pattern-lookup may occasionally be necessary in KPK even for a master, the novice has to rely on calculation for almost every decision, and then frequently decides wrongly. It may be concluded that even for so small a fragment of chess as KPK acquired knowledge dominates over calculation as a prerequisite for skilled performance. Hence it constitutes a suitable domain for the study of machine representation and acquisition of knowledge.

We shall discuss:

- (1) the "computational efficiency" of a representation; i.e. efficiency with regard to processor-time and machine memory. This comparison is made with respect to a specified machine and computer language (DEC-10 machine, Algol-60 language). What does such a representation look like for KPK?
- (2) the "cognitive efficiency" of a representation; i.e. computational efficiency with respect to the "brain machine". What does a "brain-oriented" representation look like in its machine-executable form as a computer program?

(3) how four existing machine representations for the computation of similar KPK functions compare with regard to (1) and (2) above. That is,

- i. Do they look much the same? If not, then
- ii. What does an efficient machine representation satisfying criterion (2) look like when converted into human-readable form?

Thus by identifying the representation which is most efficient in terms of goal (2) we may illustrate how masters can so easily cope with KPK and have a better understanding of what the necessary and sufficient concepts are.

The sufficient and necessary condition of a complex problem-domain is that all space-minimal solutions are time-infeasible and all time-minimal solutions are space-infeasible. Informally, the short solutions are too long-running and the fast solutions are too bulky. Such problem-domains are distinct from standard problem-domains. The latter may entail large amounts of calculation or alternatively large memories, but not to an infeasible degree. The distinction is diagrammatically illustrated in Figure 0.1.

For problems of sufficient complexity we can talk about a "human window" (Michie, 1982), - an interval on the memory-requirement axis derived from a plot of execution time versus memory space. Solution programs for complex problem domains which lie within the bounds of the human window combine two properties,

neither of which is normally taken as paramount in standard (as opposed to complex) information processing. One condition for a human-window program is that it be humanly intelligible as program text. This entails among other things that it must not be too long relative to the limitations of human memory, -- desirable for effective de-bugging of the program, for its transmission to others, and for trouble-shooting aberrant run-time behaviour (e.g. for example during suspected malfunction of an automation system). The other condition which a human-window program for complex domains must meet is that it be humanly executable, e.g. for checking purposes. This condition means that it should not be too calculation-intensive relative to limitations of human calculation capacity. This condition again has a relevance to debugging. Human-factors research has shown that the experienced programmer executes suspect modules "in the head" as an important means of fault location. At run-time under operational conditions such mental checks also have relevance during suspected malfunction. Examples of complex problem-solving where programs exhibit neither of the above properties can readily be found in, for example, computational meteorology and other problems requiring calculation-intensive mathematical modelling. But in certain application areas, particularly in control engineering, the above-stated considerations make it desirable that program text should wherever possible be both humanly comprehensible and humanly executable.

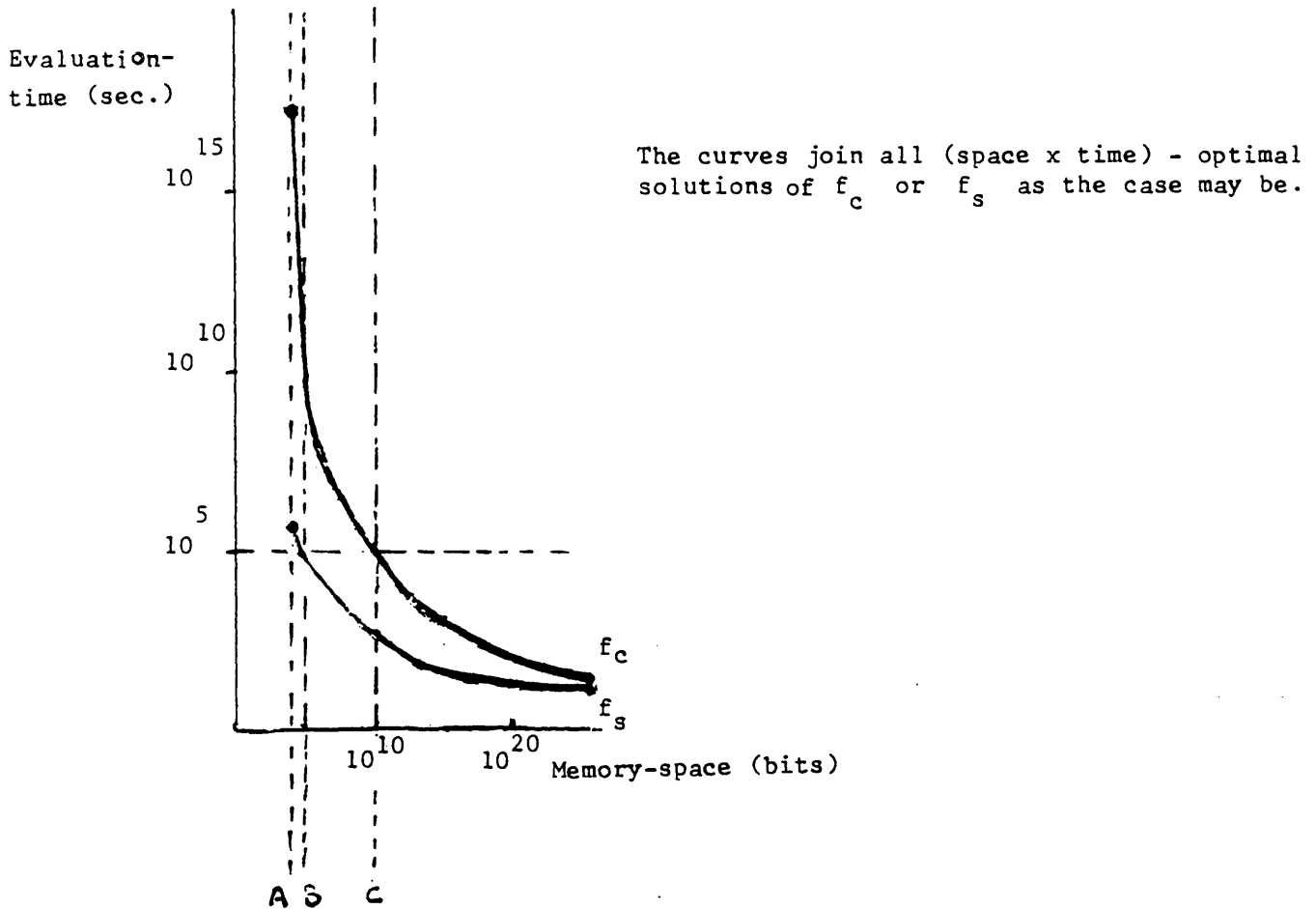


Figure 0.1

Store-time trade-off curves for two hypothetical finite functions,  $f_s$  and  $f_c$ . The maximum acceptable worst-case waiting time has somewhat arbitrarily been placed at  $10^5$  secs.

A complex problem, symbolised above by  $f_c$ , requires a far greater increment of store to convert a sub-feasible solution-program into one which is sufficiently fast-running to be feasible. The minimal required store increment for  $f_c$  in the diagram above is C - A, corresponding to 6 orders of magnitude, i.e. from approximately  $10^4$  bits store-occupancy for an ideally compact program to  $10^{10}$  bits (approximately a thousand megabytes) in our hypothetical example.

A standard problem, symbolised above by  $f_s$ , either is feasibly soluble by an ideally compact program or requires only a modest use of additional store to achieve the needed speed-up. The diagram shows a somewhat borderline case.  $f_s$  has a store-increase requirement (B - A) of ten-fold (from  $10^4$ - $10^5$  bits), possibly equal to, or more than, the maximum to which a professional programmer is willing to resort. Any lesser requirement would certainly characterise a "standard" problem.

Some of the known physical limitations on the brain viewed strictly as an information processing device are shown in Table 0, taken from Michie (1977). A given machine-feasible representation for a non-trivial problem (let us say that the representation maps all KPK positions to their game-theoretic values) may lie outside the human window in either of two directions: it may be too "intensional" or too "extensional".

By too intensional we mean a compressed representation whose complete human computation in an acceptable time (say  $10^5$  secs.) would be infeasible (remember that the calculational speed of a human is limited to about 20 binary discriminations per second (see Table 0)). The intensional solution will have few concepts (patterns) and these will have a large "grain size".

By too extensional we mean a representation which lies at the other end (the scale runs from compact representations with large grain size to bulky assemblages of very many small patterns). The extreme example would be a database which stores the game-theoretic value for each of the 98304 configurations. Here the grain size is as small as it can be. But the number of "grains", 98304, is very large, and hence the space required for their memorization is beyond practical human capacity and cannot be comprehended mentally either in this sense or in the sense of intelligibility.

1. Rate of information transmission along any input or output channel . . . . .	30 bits per second
2. Maximum amount of information explicitly storable by the age of 50 . . . . .	$10^{10}$ bits
3. Number of mental discriminations per second during intellectual work . . . . .	18
4. Number of addresses which can be held in short-term memory . . . . .	7
5. Time to access an addressable "chunk" in long-term memory . . . . .	2 seconds
6. Rate of transfer from long-term to short-term memory of successive elements of one "chunk" . . . . .	3 elements per second

Table 0. Some information-processing parameters of the human brain.  
Estimation errors can be taken to be around 30 per cent.

Sources:

1. Miller (1956) summarises knowledge up to that date. Subsequent determinations are reviewed in any modern text in physiological psychology.
2. Calculated from 1 above.
3. Stroud (1966), cited by Halstead (1977).
- 4, 5 and 6. Sources cited by Chase and Simon (1973).





To lie within the bounds of the human window a solution must require neither too much computation time nor too much memory space. If too much computation time is needed then the solution cannot be executed (e.g. for checking) by humans; if too much memory requirement, then it cannot be memorized or understood. Furthermore, representations which are not too extensional for memorization (given an effort) may nonetheless still be too memory-intensive for intelligibility from the user's point of view.

It is to the experimental examination of the propositions advanced in the above paragraph that this dissertation is directed.

When we compare a text-book representation for any domain of human expertise with a machine-feasible representation we see a distinction between information and knowledge. Any standard text on chess will have a number of concepts laid out in some ordered fashion with supporting prose and game examples. The specific examples are geared towards imparting general concepts. Thus the chess student more often learns general concepts by primary induction on examples rather than by reading text-book descriptions reinforced by trial of specific applications deduced from these. Rules comprising a human window representation are both obvious to the expert human practitioner and intelligible to the human student. Typically a KPK solution which falls within the bounds of the human window will have around twenty concepts

(rules) each representing an easily distinguishable gestalt, notion, or piece of knowledge. For the complete game of chess, knowledge-based machine representations resulting in high quality play have not yet been achieved. It has been estimated (Chase and Simon, 1973, Niveergelt, 1977) that the number of concepts to be represented as machine-implemented patterns would lie in the  $10^4-10^5$  range.

Our four test programs may be arranged as follows:

Name of program:	Harris	Bramer	Beal	Clarke
Approx. No. of concepts:	7	20	50	100000
"Grain size":	large	intermediate	small	minimal
"Transparency":				

Key: Intensity of shading indicates trend from transparent (no shading) to opaque (much shading).

We hypothesize, as indicated by the "transparency" shading of the Table, that representations of the Bramer level will prove perspicuous and manageable for people, but that departure of machine representations in either direction will be penalized by loss of this conceptual interface. If this is right, then lessons of some importance should be drawn for the engineering of socially critical software systems of high complexity (medical information systems, nuclear power stations, air-traffic control, large-scale automation, etc.).



By carrying out experiments on human chess novices using English-language translations of these four machine representations presented as open-book "cribs" (which we shall call "advice texts") we (a) obtain evidence for the existence and extent of the human window , and (b) in the process gain some insight into what a machine representation must look like if it is to fall within this window.

## 1. BACKGROUND

Before proceeding further we direct attention to Table 1 in which are collected various symbolic and abbreviative forms employed throughout this Thesis.

### 1.1 KPK from the Chess Master's Point of View

In reality, KPK is uninteresting for the chess master. It is surprising how the addition of a mere pawn to the opposing side allows us to enter a domain of much greater interest (see Piasetski, 1977). For the stronger side the entire problem domain of KPK can be generalized into the following two simple rules:

- (1) If you can run the pawn to the queening square, do so.
- (2) If you can gain the opposition ahead of your pawn, make the necessary king move to achieve this.

For the weaker side, the correct defence is simply the antidote to the above two rules. There are a few special cases which deserve mention. One occurs when the White and Black Kings are on the 5th and 7th ranks respectively, in file opposition, and a pawn advance (non-RP) to the 6th, on a non-adjacent file, achieves a winning position, e.g. WK:d5, BK:d7, WP:f5, White-to-move. Another occurs when the WK is able to get on the other side of his P before the BK can, and thereby the WK is able to

PF : The Pawn's File (always a White Pawn)  
 PR : The Pawn's Rank (always a White Pawn)  
 WF : The White King's File  
 WR : The White King's Rank  
 BF : The Black King's File  
 BR : The Black King's Rank  
 WK : The White King  
 BK : The Black King  
 P : The Pawn  
 = : Equal to  
 <> : Not Equal to  
 > : Greater than  
 < : Less Than  
 >= : Greater than or equal to  
 <= : Less than or equal to  
 | | : The absolute value of, i.e. the positive value

Table 1. Notation employed.

achieve diagonal opposition in front of his pawn, leading to a win. This is one of the few special concepts which needs to be known, for even strong players have missed the key first move in the position with White to play, WK:d1,BK:f8,WP:c3 (Clarke, 1977; see also Section 6.2.1).

However, if the precise machine representation for correct play in this ending were as simple as the rules stated above, then most of the following pages would be unnecessary.

## 1.2 History of KPK Programs

The first record of a program designed specifically to play KPK, the least mobile of chess endings, was written by S. Tan in 1972. Since then this endgame has been programmed by (among others) Larry Harris, at Dartmouth College in 1975, Michael Clarke (1977) and Don Beal (1977,1980) at Queen Mary College, Max Bramer (1977) at the Open University, Milton Keynes, Leon Piasetaki (1977) at McGill University and by my colleagues A. Shapiro and T. Niblett (1982). Clarke's program generated a database which stored the game-theoretic value and the minimax-optimal number of moves to termination for each of his computed 98304 legal configurations of the three pieces with White-to-move and Black-to-move. Beal developed the first proven correct algorithm, based on other than exhaustive enumeration, for computing the game-theoretic values for KPK. It was implemented as a decision tree in a Fortran subroutine called by a larger program used to compute other KPK

quantities of interest. Bramer (1977) employed 19 "equivalence classes" acting as goal patterns for a 1-move lookahead to try to find a "correct" move in any position where White-to-move can win. This version of the program played correctly in all critical positions (i.e. positions where White-to-play has only one winning move (Bramer, 1977)) and was known to perform correctly for very nearly the entire space of White-to-move positions. More recently Bramer (1981) reported the development of a 20-class program which plays correctly in all White-to-play positions. Piassetski's work involved a Fortran subroutine, ONEPAWN, which gives 300 "masks" encoding the game-theoretic value of every possible KPK configuration. Harris's program occupies approximately 128000 bits in core, Beal's approximately 42000 bits, and Bramer's requires on the order of 58000 bits of source code which is interpreted. For comparison, Clarke's database is stored in 786432 bits of memory.

Bramer (1980a) reported the modification of his KPK model by the addition of a further 19 equivalence classes, so that it not only found correct moves, but optimal ones. This was verified by tests with Clarke's database. Optimal programs guarantee pawn promotion in the shortest possible number of moves, while correct programs may take more than the minimum number of moves to win. Bramer's original "19-class" version was in fact optimal in 96.6% of the cases. Such "diminishing returns" effects in the programming of chess endgames will be discussed later. Bramer's is the only endgame program currently in existence which plays optimally

(as contrasted with merely evaluating the game-theoretic value of a position, or where applicable its minimax-optimal distance from safe pawn promotion). We here exclude programs which generate optimal play by direct lookup of an exhaustive database. Shapiro and Niblett were able to use the ID3 (Iterative Dichotomiser Three) program (Quinlan, 1979) based on Hunt's Concept Learning System (1956) coupled with a powerful programming tool, the CLIP parallel array processor (Zdrahal, Bratko & Shapiro, 1981) to inductively derive a set of decision rules which correctly classify all legal Black-to-move (B-T-M) positions.

### 1.3 Huberman's Work

The first success in the programming of correct play in chess endgames was reported by Barbara Huberman (1968). Her goal was to study the translation of standard textbook information on the correct play of certain elementary chess endgames to some higher level computer language (LISP). The machine-encoded versions for King and Rook vs. King (KRK), King and Two Bishops vs. King (KBBK), and King and Bishop and Knight vs. King (KBNK), were continuously revised and refined, until they seemed to play correctly in any position. Then these final implementations were assessed on their correspondence to the textbook methods which they originated from. Textbook conceptualizations were found to be sufficiently clear for this purpose, though too incomplete and imprecise for direct translation. Huberman's final programmed versions turned out to be quite successful, forcing mate within

the 50-move limit in each case. In the description which follows I have been aided by Bramer's (1977) earlier analytic assessment of Huberman's work.

The foundation for Huberman's model is the concept of a forcing tree. This model assures that from any starting position,  $p$ , where the program has the move, it will search until it finds a position,  $q$ , which satisfies a relation better than  $p$ , and reachable from  $p$  for every sequence of moves by the opposition. This process repeats until checkmate is reached (Huberman, 1968; see Fig.1.1).

Basic to this formulation are two truth functions, better and worse, which take as arguments any initial position,  $p$ , with White-to-move and a successor position,  $q$ , (at some depth) with Black-to-move. A breadth-first search is performed to find the better positions as White's goals. Any variations which lead to worse positions as a result of Black's best play are immediately rejected. The purpose of better and worse is to prune the search tree and they are in no respect complementary. Both functions are used to define STAGEs whose increasing integer value indicates proximity to checkmate. STAGEs are further subdivided into MEASUREs. The lower the MEASURE the closer we are to entering the next stage. Terminal positions in the forcing tree are all ones where Black is to move and better evaluates true. Since these truth functions are defined in such a way that White can always force a better position, and that a position,  $q$ , can be evaluated on the basis of STAGE's and MEASURE's as closer to

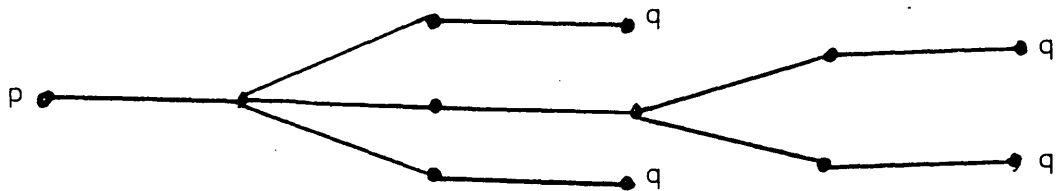


Figure 1.1 Example of a Forcing tree. The program has the move in p; it must make a move leading to a position q judged better than p for every sequence of moves by the opposition. Each iteration of the program will produce a tree like this; several iterations will be required to reach checkmate. (reproduction of Figure 1.1 plus caption from Huberman, 1968)



checkmate than *p*, we are assured the process will ultimately converge. Worse is used to avoid disasters such as stalemate or the loss of a piece, deemed too high a price to pay for attaining a given better condition. Each STAGE with its MEASUREs, supported by better and worse, was intended to roughly correspond to one textbook heuristic (rule of thumb).

In her final versions for the three endgames mentioned, Huberman requires the following combinations of STAGEs and MEASUREs:

KRK 4 STAGEs, 1 with MEASURE  
KBBK 5 STAGEs, 3 with MEASURE  
KBNK 7 STAGEs, 2 with MEASURE

However, the resulting definition of these STAGEs and the necessary extensions to better and worse to bring the search down to reasonable proportions is in each case an extremely complex combination of predicates, bearing little resemblance to book heuristics. Huberman considered the process of deciding roughly what the STAGEs are rather straightforward, closely corresponding to book information. What she found difficult were the exact definitions of STAGEs and MEASUREs, and generally it was even more difficult to define practical refinements to better and worse.

The play for KRK is quite strong (where by strong we mean correct and not far from the shortest path to victory) in the examples given. It is considerably less strong for the two more difficult endgames (KBBK, KBNK), but particularly effective in

the final stages of each endgame. Huberman paid special attention to these stages since diversions and falterings near the end could have rather detrimental effects on the effort as a whole.

Huberman's work was the first serious, practical model for the programming of chess endgames. In addition, one of the endgames successfully modelled (KBNK) is considered hard, even by human standards. Nonetheless there are some problems with her approach:

- 1) All better positions are equally good. Therefore the programs tend to make "acceptable" moves, though rarely the best.
- 2) There is little control of the depth and breadth of search, and attempts to do so necessarily led to very complex definitions for STAGEs, better, and worse.
- 3) Other than a few examples of program play, there is no indication of empirical testing.
- 4) Huberman informally proves that her programs are correct (Chapter 7) in the sense that White would eventually win regardless of how Black plays. Yet for more complex endings such proof methods would be severely taxed.
- 5) While in principle the model is generalizable, in practice it might be virtually impossible to program for more complex endings.
- 6) The model is only applicable in positions where White has a

forced win. Bramer (1977) states, "It is difficult to specify precisely the subset of positions in which White has a forced win, even in the case of KBNK". Against this, however, it might be argued that KBNK only poses problems with regard to a few stalemate positions and those where the Bishop and Knight can be forked by the Black King.

7) There is no simple a priori way of knowing that a move will actually be found by the program. That is to say, as refinements are made to the program, very precise definitions of STAGES and extension to better and worse are required to ensure that a move will always be returned.

The groundwork for programming chess endgames in a systematic, if tedious manner, was laid by Huberman. Of the efforts to program KPK, Bramer's equivalence class model bears the closest resemblance to Huberman's approach.

## II INADEQUACY OF THE STANDARD ALGORITHMIC APPROACH TO COMPLEX PROBLEM-SOLVING

### 2.1 Introductory Remarks

The work reported in this Chapter concerns an attempt to solve a complex problem, namely the value (a White win or a draw) of all KPK configurations, in the conventional algorithmic programming style. A program written in the classical or conventional algorithmic style is characterised by two properties:

1. it is relatively compact.
2. any domain-specific knowledge which the programmer may have put into the program is embedded in the bodies of the procedures in a way which is in some sense implicit only, as opposed to the clear separation of knowledge-base and knowledge-interpreter which characterises the knowledge-engineering style.

The difficulties encountered are an example of the limitations of the classical programming approach when used for complex problems, namely:

(1) proliferation of special cases each requiring its own code written to supply ad hoc fixes, these in turn leading to special cases at lower levels, and so forth;

(2) in consequence a virtually undebuggable program in which location and characterisation of errors grows in difficulty, and

(3) the introduction of cures for each local disease requires exhaustively checking for subtle maladies which may have been introduced by side-effect in other parts of the same program.

Zuidema (1974) reported that the work involved in programming by classical methods and debugging even the comparatively simple King and Rook vs. King (KRK) was enormous. For KPK our experience indicates that while the programming task by conventional algorithmic methods may be feasible in a matter of months, the debugging task is not. At the next level of complexity lies the King and Rook vs. King and Knight (KRKN) endgame which Bratko and Michie (1980) regard as not programmable by classical methods. The programming task posed by KPK appears to lie in an interval on the complexity scale which is transitional with respect to intractability of debugging, suspended, as it were, between the hard but tractable KRK level, of which Zuidema complained, and the KRKN and harder levels. The remainder of this Chapter is devoted to recording a detailed investigation of Harris' KPK program, a worthy representative of the classical algorithmic style, from this point of view.

## 2.2 Harris' KPK Program

### 2.1.1 Verification and Changes

Circa 1975 Larry Harris wrote a KPK program in Algol-60 for the Dartmouth Timesharing System (Honeywell 635). The program, like Beal's, gives as its output the game-theoretic value of any input position for both White-to-move and Black-to-move. It also

produces a test stating reasons why the position is drawn or won. J. Bitner & B. Hansche (1976) translated the Harris program into Algol-W for an IBM 360/75. After mounting the Clarke database on the disc of the IBM 360/75, these authors used it as a standard of comparison to determine the correctness of the Harris KPK Program. From their tests on the entire data-space of 98304 legal board configurations, each having a value with White and Black-to-move, the program was found to be 99.11% correct. They claim that 45% of the errors were trivial programming errors (p.7 of their report) and they give a conceptual classification and discussion of the remainder.

Part of this work has involved an effort to substantiate Bitner & Hansche's conclusions and to make necessary corrections to the program. The original Honeywell version of the Harris Algol program was initially (1977) translated into Algol-60 for the Edinburgh Regional Computing Centre's (ERCC) DEC 10 / KI 10. Then it was modified so that input was in terms of the integers 0 to 98303 in order to match the way that the database is interrogated (this will be described more fully later). Each integer in this set represents a position that was previously input in Algebraic Chess Notation. Procedures CONVALG, BASPOSNOALG, VALUES, and BASVAL were added to the Harris program to permit communication with the database, together with some additional blocks of Algol code. Numerous changes were also made in other procedures and in the main program for this purpose. This version of the program, also incorporating database interrogation, was renamed HARBAS.ALG.

### 2.1.2 Tests on 1001 Configurations

A preliminary sample of position numbers 0 to 1000 was used. All configurations in this range have the White pawn (WP) on the square a2 (where "a" represents column one and "2" represents row two on an 8 x 8 matrix), while the Kings' positions are altered. The program's value was found to compare correctly on 97.40% of this sample (See Note 1). However, closer analysis revealed the inadvertent inclusion by the program of a particular kind of illegal position, namely positions where the WP is on a2, and the Black King (BK) is on b3, which have no legal predecessor. Correction of this special case resulted in 16 fewer errors or a 0.75% improvement. To generalize this correction, all positions with the WP on the second rank and the BK in check have been marked as illegal by the revised Harris KPK program.

The 37 positions of disagreement by the "98.15%" version of the program were classified as follows:

Type 0: WP:a2, \* BK:a1 or b1, WK: legal and not next to WP;  
Black- to-move draws.

Type 1: WP:a2, WK: on the second rank and can block BK from WP;  
Black- to-move loses.

Type 2: WP:a2, WK: on the second rank, nearer to WP than BK, BK:  
on the third rank; Black-to-move draws.

\* NOTE: Piece Name: Square Name means Piece is on Square.

The breakdown of these program errors was:

---

ERROR TYPE	0	1	2
TOTALS	22	9	6

See Appendix A.1 for a discussion of how these initial Type-classed errors were corrected.

The reported 22 Type 0 errors which resulted from traversing the WK across the first 2 ranks, while the WP was on a2 and the BK was on a1 or b1. Therefore it is safe to assume that there would have been at least 4 x 22 errors of precisely this Type if the WK were traversed across all 8 rows, and at least 4 times this total, or 176, if the Pawn were tested on all 4 files in the database. In other words, more than 10% of the reported 1750 positions of error (Bitner & Hansche, 1976) were probably caused by extensions of this Type 0 error. This type of error must be considered of a trivially conceptual nature. However, if this was viewed by Bitner & Hansche as a "trivial" programming error, then at least 176/788 or 22% of the programming errors were caused by this mistake.

### 2.1.3 Tests on Random Samples

A test was attempted on a random sample of 1000 positions from the whole space employing the DEC 10 Algol-60 random number generator. It was adjusted so that a position was selected randomly from every database interval of 100 positions. Unfortunately the job could not be finished in one session because there was a system crash. However, approximately 800 randomly selected positions were successfully input to the program. For this sample there was 94.37% correctness, or in other words 45 positions were generated where the database and Harris program were not in agreement. These 45 positions of error were classified as:



Type 0.n: where the 0 implies that Black-to-move can capture the WP and "n" specifies how many moves the BK will have to make to do so. Type 0.1 positions are the same as the Type 0 positions defined earlier, except that the WP can be on any rank, rather than just the second.

Type 3: positions where Black-to-move can get to a key square in front of the WP to draw. That is, positions where the WP is not a rookpawn and the WK cannot get in front of it with opposition.

The breakdown of these error types was:

ERROR TYPE:	0	1	2	3	0.1	0.2	0.3	0.4	0.5	Not classified
TOTALS	0	0	0	5	19	6	9	2	3	1

The one position which was not classified will be held over until more such errors appear. There are two pieces of information in this set of errors which were suspect: (1) All positions are with Black-to-move (see Note 1 and later) (2) Of these, most involve a simple "foot race" to test whether the BK can reach a crucial square in time, rather than some conceptual problem. See Appendix A.2 for a discussion of efforts to eliminate these errors.

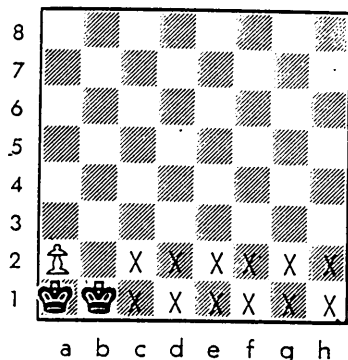
After incorporating the needed correction, a second attempt to sample 1000 random positions employing the DEC 10 Algol function SETRAN to obtain the same random number sequence, verified

EXAMPLES OF ERROR TYPES CLASSIFIED  
FOR THE HARRIS PROGRAM

---

ALL POSITIONS ARE BLACK TO MOVE

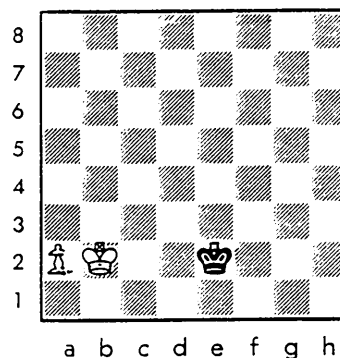
Fig. 1.



Type 0 Errors:

WP:a2, BK:a1 or b1, WK any of the squares marked with "X" are legal.

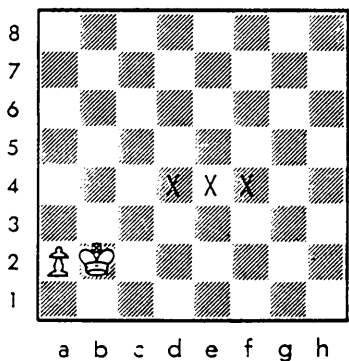
Fig. 2.



Type 1 Errors:

WK can block BK.

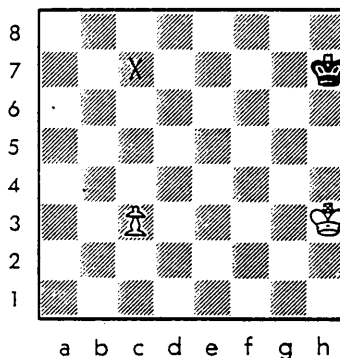
Fig. 3.



Type 2 Errors:

BK: any of the three squares marked with "X".

Fig. 4.



Type 3 Errors:

BK can reach the vital square (X) in front of WP to Draw.

the above beliefs. This time the run got as far as position number 75995, again with approximately 800 positions tested. There were only two positions of error, or 99.75% correctness. One of these was a Type 3 error which occurred earlier and was very similar to fig. 6 in Bitner & Hansche's report (see fig. 4.) The other was the position which was unclassified after the previous run. These results were encouraging because they indicated that the trivial programming errors may already have been accounted for, in addition to something extra.

Bitner & Hansche also mention several other errors in the PROCEDURE WKOPP (pgs. 8, 9). Their suggestions, with some modifications, have been implemented. Thus White can get the opposition if: 1) White-to-move can reach the square two ranks ahead of the Black King with his King. 2) The Kings are on opposite sides of the pawn and (my addition) the White King can reach a key square ahead of the pawn before the Black King can reach a key defensive square. 3) The Kings are on the same side of the pawn, and the White King is able to reach a key square on the other side of the pawn. In addition, a test for the existence of diagonal opposition has been added since this will help determine if victory can be achieved in many cases where the Kings are on opposite sides of the pawn. This correction of PROCEDURE WKOPP, coupled with the changes just described in PROCEDURE RANK234 to remove Type 3 errors, inspired confidence that the program now correctly evaluated opposition with the White pawn on the 3rd and 4th ranks. Tests with a benchmark of positions for these ranks

confirm this view.

An overview of the changes made in the PROCEDURES WKOPP and RANK234 to get algorithms which have proven to be empirically correct (that is for the B-T-M positions which were evaluated and compared with database values) points to two important features:

- 1) The decomposition of a procedure growing in complexity into several smaller, special-purpose procedures. This was done for PROCEDURE WKOPP. The basic procedure detects standard forms of opposition and whether they can be achieved. WKOPP was broken down into three conceptually different forms of opposition, with the addition of PROCEDURES KOPOP and SSOP. The first recognizes if opposition can be achieved when the Kings are on opposite sides of the pawn. The second detects whether the White King can achieve opposition by getting on the other side of the pawn, when initially the Kings are on the same side of the pawn (SSOP). In fact the evaluations, SSOP = FALSE AND KOPOP = FALSE were necessary for the proper definition of Type 3 draws in PROCEDURE RANK234.
- 2) The importance of "distance" tests to the entire domain. This technique is common to the approach of Bramer in his breakdown of the problem into 19 equivalence classes, and to that of Beal, in his 48 decision table tests. In our program, the refinement of such "distance" tests was also found to be necessary in order to correct several conceptual errors and improve the program.

Fig. 5.

WITH BLACK TO MOVE ITS A:

WPLOC= 23  
WPLOC= 24  
WPLOC= 25  
WPLDC= 26

THE CURRENT BOARD IS

-15	-13	-11	-9	-7	-5	-3	-3
-15	-13	-11	-9	-7	-5	-3	-1
-15	-13	-11	-9	-7	-5	-3	-3
99	99	99	99	99	99	99	99
16	14	12	10	8	6	4	4
16	14	0	10	8	6	4	2
16	14	12	10	8	6	4	4
16	14	12	10	8	6	6	6

WIN WK CAN GET IN FRONT OF WP WITH OPPOSITION

An example of how the Harris Program generated Type 3 errors (WK=2, BK=-1, WP=0) because the test in PROCEDURE RANK234 for whether the BK can reach the square 4 ranks ahead of the WP fails with the inclusion of d4 (marked 10). Since the Abs(10) is less than the Abs(-11), the wrong decision was reached.

#### 2.1.4 Improving Efficiency

A test on Posnums (position numbers) 1000-2000 was done on the Dundee University DEC 10 after the correct sequencing of "Closes" and "Opens" of the database file on disc. For this sample, statistics compare as follows:

	ERCC DEC 10 (K110)	DUNDEE UNIVERSITY DEC 10 (KL10)
June 19, 1977	time: 22.28	Dec. 7, 1977 time: 52.23
No. of errors	1 (cause: WPRANK=3 instead of 2)	0
Real time	Approx. 1 hour	1 hour, 3 minutes
CPU time (msec.)	4976674	1769240
Core Occupancy (words)	7679	8703

N.B. The only changes to the program HARBAS.ALG made between the two runs were those described between pgs. 20 and 24.

Most of the speed-up resulting in nearly two-thirds less CPU time consumed can primarily be attributed to the relative speed of the DEC 10 processors used (the KL10 is 2.3 times as fast as the K110). The remaining time difference was due to a programming error in "READING" from the database file.

As a result of further investigation into efficiency problems for purposes of database testing, it was found that the very costly PROCEDURE VALUES was being called several times, when its result could have been held by a variable from the initial call. CPU time savings as a result of this correction were significant. The completion of tests on the remaining 224 positions (POSNUMS 76016-97875) from the randomly selected 1000 positions required only 226 CPU seconds, or less than 3 minutes, with no new Harris Program errors found. Therefore the program had only the two errors reported for this sample of 1000 positions chosen at random

or 99.8% correctness. On the basis of this run a bigger test, not previously attempted, namely on the first 20002 positions (POSNUMS 0-10000) in the database was feasible and showed 99.1% correctness. This entire run required less than 44 minutes.

More recently (Feb. 17, 1979) it was found that only B-T-M positions of conflict (those where the database does not agree with the Harris Program) were being detected by the program. A variable which was to transfer control to the appropriate I/O section was not being initialized in the Main program. As a result, even though the values of White-to-move (W-T-M) positions were being computed by the program and looked up in the database they were not being compared.

A second run was performed on POSNUMS 0-10000 (20002 positions), but this time with conflicts on White-to-move also being tallied and output. This showed 98.33% correctness. Of the 335 positions of Harris Program error, 182 were with Black-to-move and 153 were with White-to-move.

It would seem worthwhile to mention here that all the positions in this nonrandom sample of 10001 configurations involve the WP on rank 2, as do the first 15384 entries in the database. For this run all the corrections to the Harris Program already described for such positions had been implemented, with the exception of the overhaul to PROCEDURE WKOPP, including PROCEDURES KOPOP and SSOP.

It was now possible to carry out a bigger test whose results were as follows:

<u>POSNUMs</u>	<u>TIME (hrs.:min.)</u>	<u>No. of ERRORS</u>			<u>% (W + B) CORRECT</u>
		<u>W</u>	<u>B</u>	<u>TOT.</u>	
0 - 10000	:44	153	182	335	98.33
10001 - 23627	1:00	223	306	529	98.06
23627 - 44692	1:44	175	258	433	98.97
44692 - 61285	1:05	45	66	111	99.67
61285 - 98304	1:48	50	57	107	99.71
----- Tot. 6:21		Totals: 646 869 1515			99.23

This resulted in 99.23% correctness for the Harris Program over the complete set of 98304 configurations where evaluations of both W and B-T-M positions were correctly compared with database values. A complete classification of these positions was not undertaken, but it was noted that 1378 of the positions of error involve the WP on ranks two, three and four. Enough had been learned for it to be evident that final purification of HARRIS.ALG of error, although doubtless possible in the end, was a task so little facilitated by structure and philosophy of classical algorithmic programming as to cost a rather large multiple of the amount of work which Zuidema had to expend on KRK.

#### 2.1.5 Vital Statistics

The original Harris KPK program (HARRIS.ALG), translated into DEC 10 System Algol-60, occupies 27 blocks on disc; that is, with 128 words/block and 36 bits/word, approximately 124000 bits. About 7 of these blocks are devoted to data-structures and the analysis of KPK positions. The program requires 5119 milliseconds to compute the values of one configuration, i.e. with



White-to-move and with Black-to-move. In order to access the database and perform the necessary changes in input/output, 10 blocks of Algol code (37% increase in size) had to be added to HARRIS.ALG, resulting in the version HARBAS.ALG. Another addition was a TV display routine (see Table 2 ). The disc occupancies of these various components of Algol-60 text are approximately as follows:

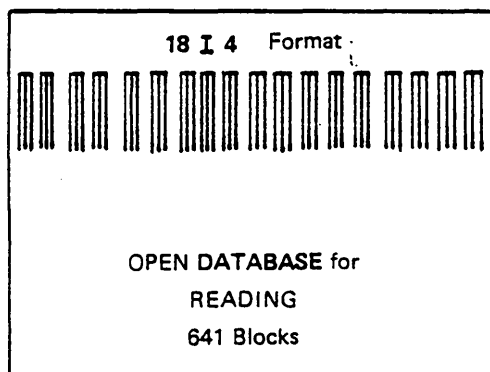
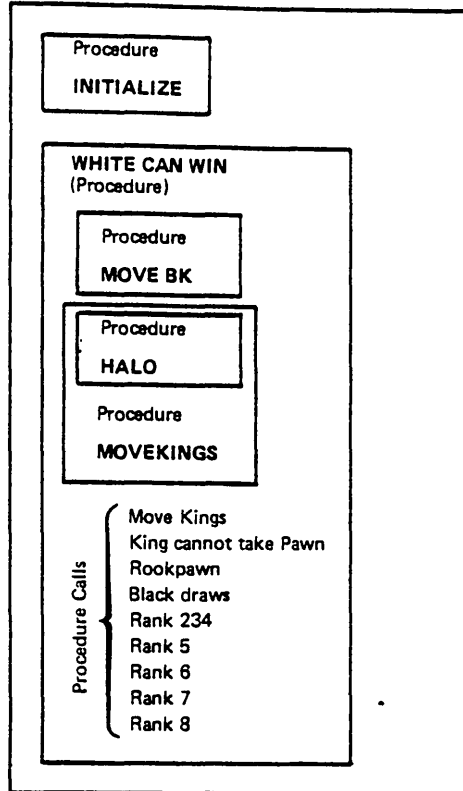
(1)	Data Structures	18000 bits
(2)	Input/Output	36000 bits
(3)	Database Access	13500 bits
(4)	KPK Routines	90000 bits
(5)	TV Display	9000 bits

---

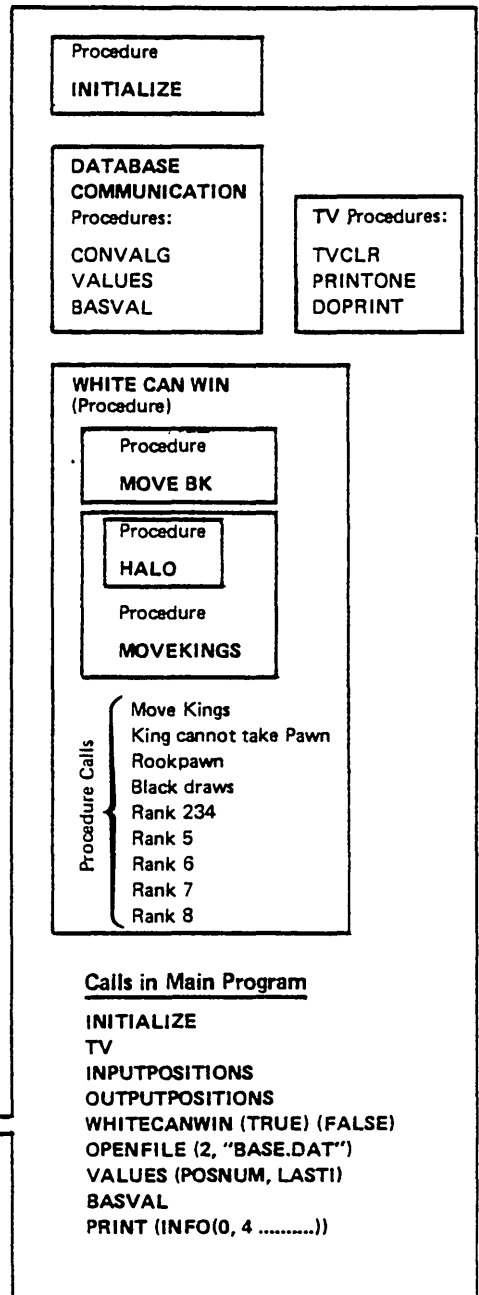
Total                    166500 bits

This version, minus the TV display procedures, occupies about 7700 36-bit words of compiled code in core. Of these, 1024 words are required by the data-structures in the procedure INITIALIZE. Based on the above figures for relative bit store on disc, we can estimate that the KPK routines require 129000 bits of compiled code. The discrepancy in size between this and Beal's subroutine, the most compact KPK algorithm, is due in part to the latter's being essentially in the form of a decision table. HARBAS.ALG requires very roughly 40 seconds on the average to compute the values of one configuration with White-to-move and with Black-to-move, access the value of the position in the data-

Table 2. HARRIS. ALG  
MAIN PROGRAM



HARBAS.ALG  
MAIN PROGRAM



base, and compare these values. Access to the database values for a configuration was sequential.

#### 2.1.6 Conclusions

The results of the work described in the previous Sections of this Chapter indicate how difficult the KPK endgame, an easy one for human masters, is to bring to finally complete and correct form by conventional algorithmic methods. We now turn to a description of three non-algorithmic approaches to KPK.

### III DESCRIPTION OF THREE NON-ALGORITHMIC APPROACHES TO KPK

#### 3.1 Beal's KPK

This work is significant for two reasons in particular:

- 1) it was the first correct program to be validated exhaustively against a pre-computed database for any chess endgame;
- 2) the information provided by a KPK database (Clarke, 1977) was used to develop a correct routine in about two weeks (contrast experience with HARRIS.ALG).

The correct Fortran subroutine was devised by performing a series of tests on geometric distances (such as `max (filedist,rankdist)`) between various pieces and squares. First these tests were applied to White-to-play positions to determine their game-theoretic value (win or draw) based on the applicability of one of 48 decision rules. Then Black-to-play positions were evaluated by a 1-ply lookahead mapping into the value of successor positions with White to play.

Beal's technique to generate a correct program was to systematically compare each current version with the KPK database until a number of misclassifications had been accumulated. These were studied, and rules were devised or modified to bring about correct classifications. Furthermore, the study of erroneous rules or positions which lacked classification altogether, permitted the intuitive generalization of rules for similar patterns. Such pattern-oriented classification techniques have already been mentioned in facilitating the correction process for

the Harris program. In chess endgames, patterns of considerable generality are often easy enough to ascertain because the particulars of a position are less important than the underlying motif. Beal states:

"... other workers have spent a great deal of time and effort transforming this kind of information into a computer program and it may be that a large portion of this effort lies in a possibly unsuspected quarter -- that of being their own 'devil's advocate' when considering the correctness or otherwise of parts or proposed parts of their algorithm."

The use of empirical proof techniques (i.e. a database for verification), coupled with the simple and uniform decision-table representation, served to speed up the correction process immensely. Accompanying this approach was the fact that new rules could be added without much consideration for their redundancy or assimilation to existing ones. Nor was there any effort to make these descriptions intuitively meaningful to the human player. Correctness and speed with regard to discriminating wins from draws were the only criteria in Beal's work. In achieving this he discovered, as Zuidema anticipated (Zuidema, 1974), that the number of rules (tests) increases disproportionately as the number of exceptional cases (misclassified positions in Beal's case) decreases (see Table 5.2).

While Beal's KPK subroutine does not play moves, he states that it could easily be constructed to do so by using the functions KPKWV and KPKBV, which calculate position values with White

and Black-to-move, respectively. However, the mapping into appropriate rules to ensure correct play, as opposed to just value-preserving play where cycling in won positions might occur, would require additional heuristics. He orders these as follows:

- 1) If there is only one winning move, make it.
- 2) If advancing the Pawn preserves the win, advance it.
- 3) Otherwise
  - (a) select the King move which advances the King a rank.
  - (b) If there is choice in such moves, play the King move which minimizes K and P file difference.
  - (c) If there is still a choice, play the King move nearest to the edge of the board.

The decoding of Beal's 48 rules, which have been generated in no particularly ordered fashion, into an English language translation (see Chapter V) proved to be no mean task. As Beal admits, "Greater chess expertise would probably have helped" towards the production of more economical rules. However, the task of deciphering Beal's "well-encoded" rules may be considered analogous to the difficulties encountered when an expert programmer tries to debug a novice's unnecessarily complex version. In each case there is the problem of translating a bitty into a well-conceptualized representation.

Some examples of the implementation of these rules appear in Section 5.3.2.

## 3.2 Bramer's KPK Programs

### 3.3.1 Goals

Bramer aims at a well-conceptualized system of knowledge representation for elementary chess endgames. Two major criteria underlying his choice of model were:

- 1) The algorithm constructed using the model should be natural from the viewpoint of a chessplayer and commensurate with his view of the complexity of the task.
- 2) That the algorithm should be capable of refinement in the light of experience in a manner which preserves the previous property.

(Bramer, 1977)

A long-term goal of his research is the development of a representation suitable as the basis for a fully automatic system of algorithm refinement. He clearly parts from the usual approaches based on tree-searching and numerical evaluation functions. He feels that the equivalence class model tested on the endgames KRK (King and Rook vs. King) and KPK have been successful in adhering to the two above criteria, and that the long-range goal, automatic algorithm refinement, is not out of the question. A full description of Bramer's equivalence class model can be found in his Ph.D. thesis (Bramer, 1977), so we shall just present an overview of the model and its implementation for KPK.

### 3.3.2 The Model

The idea of equivalence classes is that human chess players learn elementary endgames through a few examples given on a few pages in a textbook. The human generalizes these examples into patterns which do not change in concept when positions are shifted by a few files and/or ranks. Still he is able to recognize exceptions to familiar-looking situations, and his prime consideration is, "What is the best move which will bring me into a position which either is a known goal pattern, resembles one, or decreases the 'mental distance' from one?" These goal patterns, with regard to only one side to move, each representing the unique set which every position must fall into, are equivalence classes. Equivalence classes are the subsets which collectively completely define a set and are mutually exclusive.

The basic model starts by generating all legal successors with one side to move. The side chosen is the stronger side (in this case White) since its correct play is more interesting and better prescribed in textbooks. Then the most favourable of these is chosen. Thus a general means of evaluating all positions with Black-to-move must be developed. The model encompasses no tree-searching below the level of the immediate successors of a given initial position. This format has been sufficient for the two endgames tackled (KRK, KPK) and is consistent with the belief that such pattern-knowledge rather than analysis is employed by humans for elementary endgames. For more complex endings more tree-searching would undoubtedly be needed.



Once a set of all legal successors, that is, all positions, B, with Black- to-move starting from an initial position with White-to-move, have been generated, they are broken down into subsets (equivalence classes) whose membership is unique and exhaustive, i.e. every position in B can only belong to one subset and there must be a subset for every position. These equivalence classes having been defined, it is then necessary to order them in such a way that it is possible to say that every member of one class is better (in the sense of Huberman) than every member of another class or vice-versa. This ordering is in general possible only when all the positions in a class have some very strong feature in common, related to the particular endgame under consideration. These classes can be defined statically, without any forward analysis. The position with the highest ordering (class value) is then chosen. In the event that several successor positions fall into the same equivalence class, further discrimination can be performed by applying appropriate associated functions which are analogous to Huberman's measures. However, these associated functions may determine position value by further minimizing or maximizing an argument. There may be several back-up associated functions which can be applied in the event of further ties in position value. Once the equivalence classes and their associated functions have been defined in a sufficiently precise way, we can be confident that the model will play an elementary endgame correctly, since this means play from all classes of positions in that ending has been considered.

### 3.3.3 Implementation of the Model for KPK

Bramer's first version for KPK contained 15 equivalence classes. These classes broke down the problem of KPK into some larger overall categories such as "stalemate", "pawn can be immediately captured", and "pawn can run", as well as more specific ones such as "the BK can occupy the square in front of the pawn", "the WK is on the square in front of the Pawn and Black can take the opposition", etc. In refining this coarse version of his algorithm, Bramer found it necessary to treat the RP as a special case. Despite this also being the treatment of RP in standard chess texts (Fine, 1941), Bramer had hoped he could get around this. The 15 class model provided a good testing ground for learning the modifiability of his algorithm.

Since Bramer's long-range goal was perfect play (that is optimal play as opposed to just correct play, see Bramer, 1980a), he chose a unique sample by which the program's play could be studied. These were the 1733 positions where White can only play one move which wins, positions where the "pawn can run" being excluded. Of the 81662 legal positions with White-to-move (not including P on 8th rank), only 62480 are wins, and 47223 of these wins are "trivial" because the pawn is essentially free to run. Therefore 15257 wins are non-trivial, and these 1733 would seem a good sample from them. The method used to refine the algorithm was simply "exception reporting". That is, program moves were compared with those in the database, and positions where they disagreed were output. In this, the initial version of the algo-

rithm, there were 193 exceptions, all of which were corrected via a lengthy, but straightforward, series of amendments to the equivalence classes, their value table, and their associated functions. The analysis of exceptions enhanced the understanding of how classes might be simplified, combined or generalized. The breakdown of the problem-space into equivalence classes facilitated the debugging process since it was always fairly easy to discern which class or associated function was responsible for an error. This was in contrast to the debugging of the Harris program with its complex procedures.

The final algorithm consisted of 24 equivalence classes. After further analysis of the selection table for this algorithm, it was asserted that a number of these classes were in fact unnecessary, as they could be "ORed" together. Thus six classes (16, 17, 21, 22, 23, and 24) which were related in that they all had the predicate "Rookpawn", were combined into one class, 16. As with the Harris Program, each time a change was made to the algorithm it was necessary to retest the entire position space before one could be certain of its effects. Recalling that one of Bramer's goals was that his model closely corresponds to textbook descriptions, we find that each of the 19 classes satisfies that criterion, and that in some cases textbook knowledge must be expanded or further generalized.

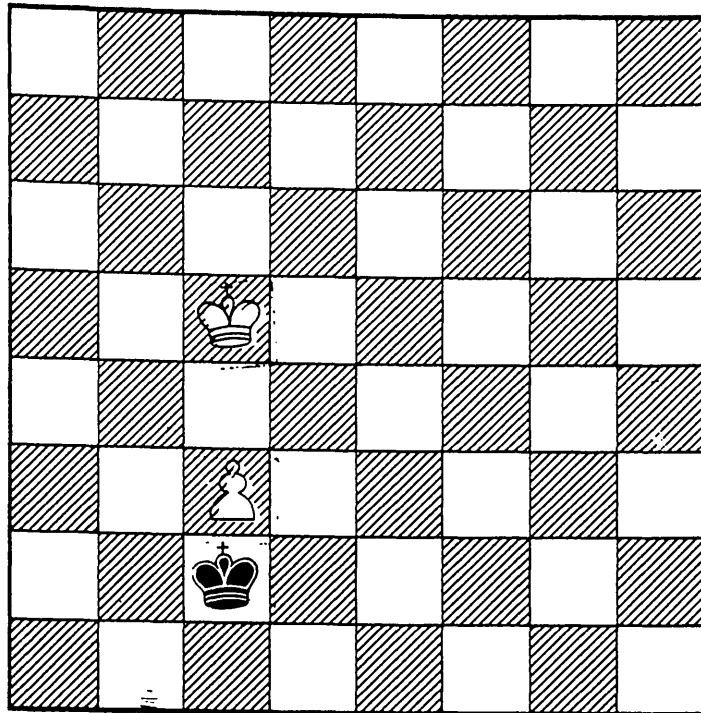
After the above refinements plus a few more, the "19 Equivalence Class Model" played correctly in each of the "1733

critical" positions (for an example see Section 5.3.5), although not in every position of the entire space. It returns optimal moves in all but 2139 of the 62480 positions where White-to-move wins, or in other words it plays optimally in 96.6% of the positions. The fact that it also played correctly in all the 1733 positions tested is not surprising, since all "hard" KPK wins would be expected to fall into one these 1733. In 921 (43.1%) of the non-optimal cases the incomplete implementation of "Pawn can run", which was Class 4, was held responsible. A 20-class model subsequently gave correct play in all cases (Bramer, 1981).

Bramer (1980a) reports the successful modification of the 19 equivalence class model for KPK by the addition of 19 new classes to generate an algorithm which plays optimally for every KPK position. Again we see this "diminishing returns" effect mentioned earlier. For a relatively small number of exceptions it was necessary to do a great deal of work. I have reported the same phenomenon in attempting to correct the Harris KPK program, as also did Zuidema (1974) for KRK. A main overall conceptual framework seems to cover most of the positions in the problem-space, but then the special cases may require that the number of rules or classes be doubled. The implementation of the equivalence class model for optimal KPK play is the same as for correct play, except that now there are 38 classes and 13 associated functions. The analysis of "effective distance" (i.e. the real distance between two squares in King moves, taking into account necessary detours (Bramer, 1977a)), which is embedded in

Fig. 6.

White to move



(This is Fig. 7 from Bramer (1980a).  
White's optimal move is 1.Kc4, winning  
in 13 ply.)

the subroutine "Pawn can run", was helpful in bringing about an optimal program.

### 3.4 Clarke's KPK Database

#### 3.4.1 Construction and Organization

As alluded to earlier, if we consider that the Pawn can be on four files (symmetry across the midline) and seven ranks, and the Kings can be anywhere on the chessboard, that gives:

$$7 \times 4 \times 64 \times 64 = 114688 \text{ configurations for KPK.}$$

Reducing this number for PR=8 with White to move, we deduct 16384, leaving 98304 legal configurations with each side to move in the state space. Each position in this state space can be uniquely represented by the integer 0 to 98303 returned by the formula:

$$I = 16384 \text{ WPR} + 4096 \text{ WPF} + 512 \text{ WKR} + 64 \text{ WKF} + 8 \text{ BKR} + \text{BKF} - 37449,$$

where,

WPR is the White Pawn's rank (range 2-7)

WPF is the White Pawn's file (range 1-4)

WKR is the White King's rank (range 1-8)

WKF is the White King's file (range 1-8)

BKR is the Black King's rank (range 1-8)

BKF is the Black King's file (range 1-8)

The complete table for KPK was constructed by the technique of breadth-first backing up from terminal positions. Terminal positions are defined as those where White immediately wins (because

P has queened) or Black draws (because he is stalemated). For Clarke's KPK these have all been defined with regard to Black-to-move. Depth zero losses (of which there are 12985) are those where PR (pawn's rank) equals 8 and B does not attack the P, or PR equals 8, B attacks the P, and W defends it. Depth zero draws are defined by either: B can take the P on the move (10093 positions) or Black is stalemated (9 of these). Depth zero wins and losses with White-to-move are defined by their transposability and necessity to transpose, respectively, into depth zero positions for Black. For each KPK configuration, I, a twenty-bit word consisting of an eleven-bit part W(I) -- one bit for each of the ten possible White moves in I plus an "evaluated" marker bit -- and a similar nine-bit part B(I) for Black (Clarke, 1977). A more complete account of the breadth-first backing up, as well as a fuller description of this program, can be found in the above reference, but we shall just give the essentials considered necessary for understanding how Clarke's database is used in conjunction with the other KPK programs under discussion.

After defining all the terminal positions we "back up" to all predecessor positions until all positions have been evaluated or left as members of the scanned-but-unevaluated group. These are positions which will transpose into each other after some looping, e.g. the drawn positions where Black can maintain the opposition indefinitely until White finally gives up his pawn or delivers stalemate.

As mentioned, the value of each KPK configuration is stored in a 20-bit word. The depth of draw or loss is held in W(I) for W-T-M positions and in B(I) for B-T-M positions. These depths range from 0 to 26 for both W(I) and B(I). Hence f(I), the figure which actually appears in the database, is defined by:

$$f(I) = 27 * W(I) + B(I)$$

and can range from 0 to 729. Wins and draws break down as follows:

<u>DEPTH</u>	<u>W(I)</u>	<u>RESULT</u>	<u>B(I)</u>	<u>DEPTH</u>
0	0	Illegal, Black in Check		
	1	Draw by Repetition	0	
0-5	2-7	Black can take Pawn	1-7	0-6
0-18	8-26	White Wins	8-26	1-19

In actuality, only 262 of these values occur, and only 252 more than once. In appearance, the database is Fortran format 18J4 (18 columns with allocation for up to 4 digits each) and contains 5462 such lines. Since each of the 98304 configurations requires 8 bits of store, the database requires a total of 786432 bits of memory. One complete example of how the value of a KPK position can be retrieved from the database is given in Section 6.2.1.



Recently Shapiro and Niblett (1982) have pointed out that Clarke's database includes 384 B-T-M positions which have no legal predecessor (e.g. WK:c1, WP:a2, BK:b3). The Harris program recognizes such illegal positions and classifies them as such. There is also at least one position with the P on the 8th rank, Black-to-move (e.g. WK:a3, WP:b8, BK:a1) which is misclassified because stalemate has been overlooked.

Of the two other KPK programs studied in this thesis, Beal's would not have been affected by the latter error in the Clarke database since it did not attempt to evaluate positions with the P on the 8th rank, while Bramer's would have overlooked this additional stalemate.

IV COMPARING THE FOUR REPRESENTATIONS  
ON COMPUTATIONAL EFFICIENCY

4.1 Their Implementation on the DEC 10

The four programs under discussion, Harris', Beal's, Bramer's and Clarke's, have all been implemented on the Dundee University DEC-10. Each has been translated into Algol-60, with the exception of Clarke's database which was generated by a Fortran program, and received on a magnetic tape. Instead we do sequential lookup and input-output in Algol on Clarke's database. The implementation of Harris' program has already been fully described in Section 2.2. The Beal program was implemented as a direct translation from the original version in Fortran (Beal, 1977) to Algol. The only modification was the addition of a few lines of code for Input-Output.

The process of implementing Bramer's program was somewhat more problematic. Essentially the program was divided into three parts. The first part consists primarily of the rather long Procedure CANRUN which was translated from the Fortran version (Bramer, 1977a). In addition, we received Algol versions of the Procedures LEGAL and STALEMATE which are prerequisite for further tests on KPK positions. The next part consists of the Procedure FINDROW (Bramer, 1980a) which determines one of 38 classes which a successor position with B-T-M will fall into. Then the Procedure CLASSVAL orders these 38 classes, just as Bramer does in

the work referenced above. The third part is mainly comprised by the Procedures MAKEMOVE, GENMOVES, BESTMOVE, TIEBREAK, ASSFNVAL, and READTABLE. MAKEMOVE defines the possible K and P moves in KPK. GENMOVES tries these moves for a W-T-M position, calls up LEGAL to test for legality, and determines a class for the legal successor positions by calling FINDROW. In the event that two or more successor positions fall into the same highest valued class, TIEBREAK is called by BESTMOVE. TIEBREAK in turn calls ASSFNVAL which computes the associated function values for the tying moves. READTABLE is used simply to read in the numbers and orders of the associated functions from a file (FUNC.NUM) where they are stored. TIEBREAK is continuously called to bring in the next associated function for these tying moves until the tie is broken or until the associated functions for that class (at most four) have been exhausted. If the latter occurs, then the tie is arbitrarily broken by choosing the lower (lowest) indexed K move. Thus far for all positions tested it has been unnecessary to choose this arbitrary tie-breaking rule. This should be expected since, if the program is to play optimally, then there can be little room for random decisions. The last part of Bramer's program was mostly designed from our understanding of Bramer's model as published (Bramer, 1977, 1980a) but not directly copied or translated from any actual program listing. The third part also consists of a very short Procedure MOVES and a few lines in the MAIN program for purposes of I/O.

The Clarke database is accessed independently in Algol by a program called BASE.ALG. This program is entirely composed of procedures extracted from HARBAS.ALG which, as described earlier, was used to compare Harris Program values with database values

for a given position. Again the Procedure VALUES is used to access the database value for a position number, or range of position numbers from 0 to 98304. Here the Procedure BASVAL has been slightly modified and improved to output more specific information on the minimax-optimal value of a position. The Procedures EXT, POS, BASPOSNOALG, and CONVALG are used for conversion of a position number (POSNUM) to Algebraic Notation for the purposes of output. The MAIN program also includes a few lines of code used to keep a record of the time consumed and the time differential in this sequential process of database access.

#### 4.2 Spatial Factors

With regard to core and disk requirements in terms of space the programs compare as the following table shows:

<u>Name</u>	<u>Blocks (Disk)</u>	<u>Words (Disk)</u>	<u>Words (Core)</u>
BASE.DAT	640	81928	--
BASE.ALG	7	910	3583
HARRIS.ALG	30	3822	5631
BEAL.ALG	9	1207	3583
BRAMER.ALG(38)	31	3948	7167
BRAMER.ALG(19)	21	2688	4607

This table shows that running the Beal Program is comparable to running the database access routine (BASE.ALG) in terms of words required in core. Beal's decides which of 48 rules is applicable for a given position in determining its game-theoretic value. BASE.ALG decodes a value for a position in the database into its game-theoretic value with each side to move, checking for legality, and converting the input, which is a position number, into output in algebraic notation. The Harris and Bramer programs are considerably bigger in core. However, readers are

reminded of the varied knowledge offered by these programs -- that is, the Harris Program provides a reason for why it has decided a position is a win or a draw, and the Bramer Program, while only useful in evaluating White-to-move positions, also provides the best move (38-class version) or winning move (19-class version).

#### 4.3 Time Factors

Table 4.1 shows that the Beal Program and BASE.ALG are also similar with regard to time factors, again demonstrating the computational efficiency of the decision table approach. The Harris and Bramer programs are also similar with regard to CPU time consumption; Bramer's runs slightly slower (.18 sec./pos. avg. vs. .15 sec./pos. avg.), but in compensation:

- 1) It is correct;
- 2) It gives optimal moves in W-to-play wins positions.

For example, in position number 10000, W to play, given in Table 4.1, Bramer's Program gives the optimal move, WP:c2-c4, whereas Harris' returns: "White wins, WK can get two ranks ahead of WP"; while this is true, and the procedure of bringing the WK two ranks ahead of the WP would lead to a win, it is certainly far from optimal.

Table 4.1

Statistics on 4 KPK Programs for same input positions

Position Number	Algebraic Representation	Access time (sec.)	BASE.ALG		HARRIS.ALG	CPU sec.	BEAL.ALG	CPU sec.	BRAMER.ALG*	CPU sec.
			W: B:	W: B:						
1000	WP:a2,WK:h2,BK:a6 B:	.40	DRAW in 4 moves DRAW in 4 moves		DRAW, BK can capture WP DRAW, BK can capture WP	.14 0	0 (DRAW) 0 (DRAW)	.06		
1001	WP:a2,WK:h2,BK:b6 B:	.39	DRAW in 4 moves DRAW in 4 moves		DRAW, BK can capture WP DRAW, BK can capture WP	.16 0	0 0	.06		
∅	WP:a2,WK:a1,BK:a1 B:	.08	Illegal Illegal		Illegal Illegal	.08 0	1† 0	.06		
80000	WP:d6,WK:e5,BK:a8 B:	22.90	WINS in 2 moves LOSES in 2 moves		WIN, WP to Queening Square DRAW, BK can achieve opposition even with WP‡	.16 -1	1 (WINS) -1 (LOSES)	.05	WP:d6-d7	.18
85555	WP:a7,WK:a8,BK:d7 B:	24.52	WINS in 2 moves DRAW in 1 move		WIN, WK can block BK DRAW, BK can pin WK to P	.14 0	1 0	.06	WK:a8-b8	.12
10000	WP:c2,WK:e4,BK:a3 B:	2.99	WINS in 8 moves LOSES in 11 moves		WIN, WK can get 2 ranks ahead of WP WIN, WK can get 2 ranks ahead of WP	.16 -1	1 -1	.06	WP:c2-c4	.18
40000	WP:b4,WK:b7,BK:a1 B:	11.51	WINS in 5 moves LOSES in 5 moves		WIN, P can win race to Queening Square WIN, P can win race to Queening Square	.27 -1	1 -1	.06	WK:b7-c8	.22
20001	WP:a3,WK:a8,BK:b5 B:	5.84	DRAW in 2 moves DRAW in 2 moves		DRAW, BK can capture WP DRAW, BK can capture WP	.14 0	0 0	.06		
20002	WP:a3,WK:a8,BK:c5 B:	5.79	DRAW in 3 moves DRAW in 3 moves		DRAW, BK can capture WP DRAW, BK can capture WP	.14 0	0 0	.06		
Words (in core)			3583		5631			3583	7167	
Average (CPU TIME) (secs.)			.05		.15			.06	.18	

\* W to move WINS positions only.

† Does not test for legality.

‡ Wrong answer.

Note: Each entry first shows the value of the outcome, WINS, DRAW, ILLEGAL, from the point of view of the side to move. The value is followed for BASE.ALG by the minimal-optimal path length to the given result, reached by pawn-promotion for WINS or LOSES, and pawn-capture, leading to repetition or Stalemate for DRAW.

For HARRIS.ALG the result as output by the program, from White's point of view (WIN, DRAW, ILLEGAL) with accompanying reason is given.

For BEAL.ALG White-to-move wins is represented by 1, draws by 0, and Black-to-move loses by -1.

For BRAMER.ALG the program's winning move for White is given.

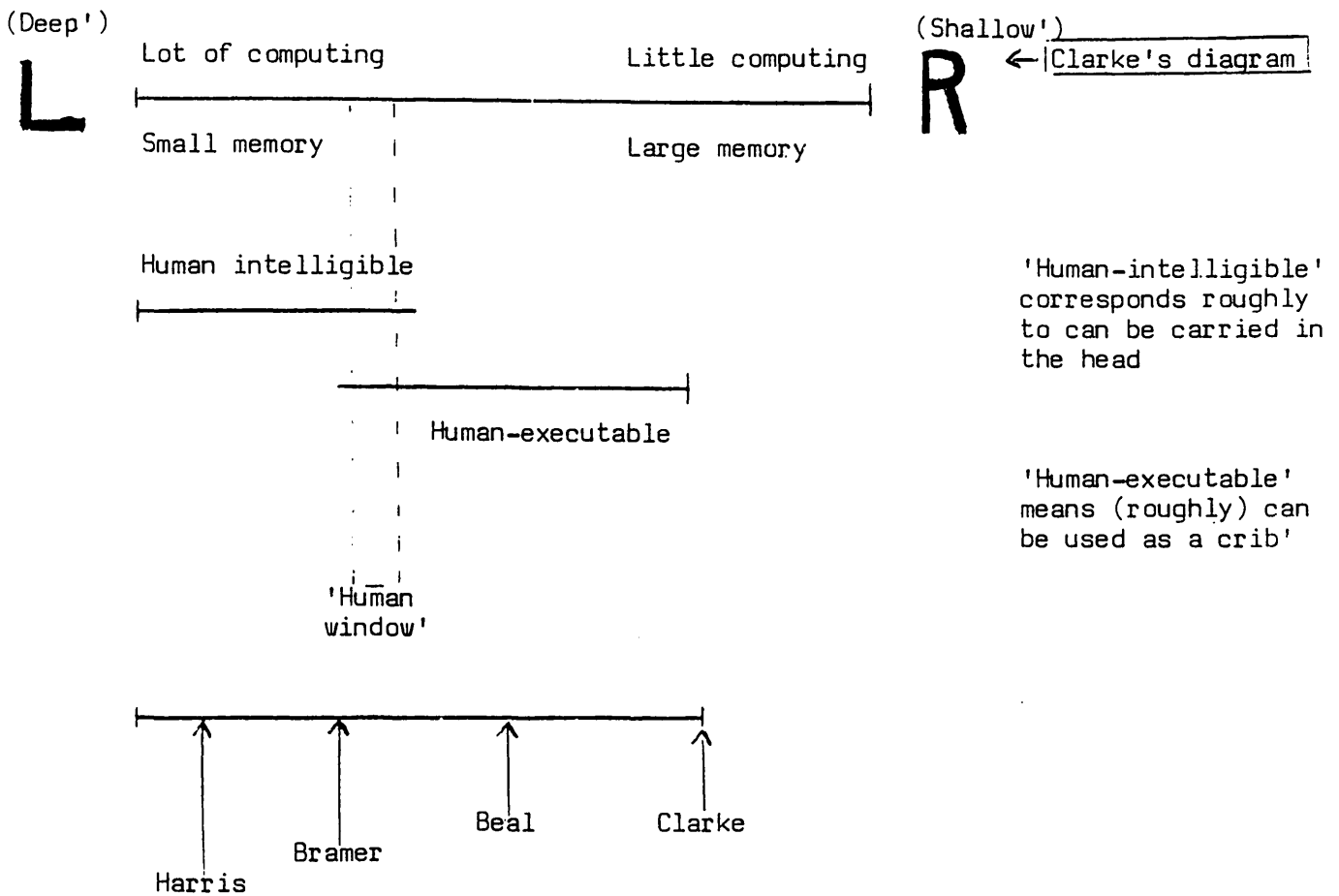
Average CPU time for BASE.ALG is taken from average access time between two adjacent positions.

## V EARLY EXPERIMENTS

### 5.1 Objectives

In the previous section we compared our four specimen programs on computational efficiency. In providing a "common denominator" for the testing of these programs, i.e. the Algol-60 language on the DEC 10 computer, we have attempted to maintain as closely as possible a resemblance to their original form. That is, changes due to translation into Algol-60 or to meet DEC 10 system specifications have been carried out in the spirit of maintaining the program's original format wherever possible. Input/output encoding has therefore been kept to a minimum.

Now we shall turn our attention to the "cognitive efficiency" of these four specimen programs. Which representation, when translated back into English, will serve as the most effective "advice text" for human beginners? That is, which advice text will prove to be most intelligible from the human user's point of view? (See Figure 7 for our hypothesis) Hence we now embark on the translation process of these three programs (of course it would not be practical to attempt to translate the database) into some suitable tabular, English-language representations. Beal provides us with a decision table of 48 rules for his program (Beal, 1977, 1980). We attempt to modify it, if possible, into a more usable English-language decision table. At first this may seem like a rather cumbersome and futile exercise, unnecessarily enlarging the outward appearance of the table. However, we hope that the effort expended in this direction will pay dividends in the clarity provided for the naive human user. The Harris Program



**Figure 7.** The four representations which are to be treated as 'cribs'. Top half of figure taken from Clarke (1977) bottom from Michie (1982a).



is then flowcharted, translated and organized according to procedural flow. Lastly Bramer's "19 Equivalence Class Model" is translated and organized into a similar decision-rule structure as Beal's, mapping positions into Classes, Classes to their appropriate Associated Functions, and finally providing a correct move in W-to-play wins positions.

## 5.2 DESIGN OF THE EXPERIMENT

Each subject was handed the given Advice Text and seated at a table equipped with an empty chessboard and the three pieces WK, WP and BK together with scratch paper. Preliminary experimentation was performed on four subjects averaging 1628 in rating. Three of the four were tested on the Beal Advice Text alone, and the fourth was tested on the Harris-Kopec Advice Text as well (see Table 5.1). The Bramer Advice Text was still being prepared and hence not used in this pilot experiment. The results of the first two subjects tested, Beveridge and Feather, were of particular interest since Beveridge had a high rating for the purposes of this experimentation, and some computer science coursework as well, while Feather was a computer science Ph.D. student. Beveridge completed only 9 of the 15 stimulus test positions in the hour allotted, but all of his responses for Rule Number and Value were correct. Feather completed 10 positions, scoring 8/10 on Rule Number (R) and 9/10 on their corresponding Values (V).

The actual experiment, was performed according to the same conditions as in the preliminary experimentation as described above. The decision table, definitions, and the 48 translated rules appear on the pages which follow the test positions, thereby comprising the Beal Advice Text (see Section 5.4.1 for further

Table 5.1

EXPERIMENTAL DESIGN

<u>Pilot Subjects</u>			<u>Treatments</u>
Name	Grading	Age	Advice Text Used
1. A. Beveridge	1890	19	(1)
2. M. Feather	1600*	25	(1)
3. P. West	1500*	33	(1)
4. H. Barrie	1520	15	(1,2)
Average:	1628	23	

<u>Experimental Subjects</u>			<u>Treatments</u>
Name	Grading	Age	Advice Text Used
1. H. Borland	1625	62	(1,2) <sup>¶</sup>
2. A. Mountford	1625	26	(2,3) <sup>††</sup>
†3. R. McDonald	1480	25	(1)
4. H. Urquhart	1535	16	(2,1)
5. D. Ward	1725	39	(3,1)
6. B. Ratcliffe	1365	19	(1,2)
†7. E. Campbell	1580	25	(3)
8. R. Kelly	1735	19	(2,3)
9. T. Lacey	1485	30	(1,3)
10. N. McGregor	1600*	16	(2,3)
11. L. McGregor	1680 <sup>†</sup>	16	(1,2)
†12. F. Taylor	1610	28	(3)
13. J. Blaikie	1250	39	(2,1)
†14. R. McKay	1300*	33	(3)
Average:	1542	28	

KEY

(1) = Beal; (2) = Harris-Kopec; (3) = Bramer

\* Rating is an approximate estimate.

¶ No answers given; subject not used in main tabulations for this Advice Text.

†† Subject misunderstood the task, and therefore his results with this Advice Text will not be tabulated.

† These one-treatment tests were regarded as incomplete, and have been excluded from the main tabulations.

details). Experimentation with the Harris-Kopec and Bramer Advice Texts was designed accordingly as described in Sections 5.4.2 and 5.4.3 respectively.

The experiment was comprised of 14 subjects averaging 1542 in rating, 28 years in age (Table 5.1). Ratings ranged from 1250 to 1735; ages ranged from 16 to 62. Of the 14 subjects' results, only 10 were considered usable as "matched-pairs results", where a subject was tested on two Advice Texts. Subjects were given one hour for each Advice Text, with a 20 minute break in between. Two of these paired-results involved attempts to use an Advice Text where there was a complete misunderstanding.

### 5.3 Method: The Translation Process

#### 5.3.1 Translation of Beal's Program

Despite the presumption that Beal's program had been completely and correctly translated from Fortran into DEC-10 Algol (Section 4.1), the process of translating Rules and Entry Conditions from Algol into English demonstrated otherwise. The Algol program had compiled and executed, giving the correct answer on all input positions thus far tested. Nonetheless the necessity of identifying each Rule in the program in order to produce a precise English-language "Advice Text" resulted in the discovery of a number of transcription errors in the Algol code. In fact, errors in Beal's decision table for Rules 6, 31, and 33 (Fig. 3, Beal, 1980) which do not appear in the Fortran listing of the same publication, (see Note 2) were also discovered during this translation process.

Many rules are somewhat redundant or repetitive. Some implicit entry conditions are not specified, i.e. Rule 12, which has as entry condition:  $PR=7, WR=8$  gives  $(W \rightarrow PP)=2$  &  $(B \rightarrow Q)=0$ , White wins, meaning that the WK's distance from the P's effective rank (effective rank equals PR, except when PR equals 2, then 3) equals 2 AND the BK is on the Queening Square, then White Wins, assumes  $PF \langle \rangle 1$ .

The position space for Beal's program regards files 1-4, ranks 2-7 as unique for the P. The two Rules which suffice for most of the position space, numbers 7 and 8, would correspond to Harris' and Bramer's (Class 4) "Pawn-Can (Cannot)-Run" and "King-Can (Cannot)-Take-Pawn" (Class 1 Bramer) routines. As Table 5.2 shows, these two Rules alone satisfy 71.8% of the KPK configurations. Rules 7, 8, 28, 31, 4, 33, and 5 together handle 90% of the position space. If one goes over the Rules, it is quickly apparent that for the most part they are very specific. There is much room for combining, modifying and in effect deleting many of them.

Table 5.1

RULES vs. PERCENT OF SPACE

<u>Rule number</u>	<u>Number of Rules (total)</u>	<u>Incremental Percentage of space satisfied</u>
7	1	$\frac{54336}{85923} = 63.2$
8	2	$\frac{61716}{85923} = 71.8$
28	3	$\frac{66493}{85923} = 77.4$
31	4	$\frac{69962}{85923} = 81.4$
4	5	$\frac{73032}{85923} = 84.9$
33	6	$\frac{75559}{85923} = 87.9$
5	7	$\frac{77348}{85923} = 90.0$

The above table shows that 7 of Beal's 48 Rules covered 90% of the problem space, and that the remaining 41 rules were necessary for "exceptions".

Note: Figures used are taken from Beal (1977).

### 5.3.2 Examples of Beal's Decision Table Approach for KPK

Ex. 1. Rule 7, which detects 54336 of the 98304 possible configurations, is:

$$(B \rightarrow Q) > (PP \rightarrow Q)$$

where the symbols are defined as follows:

B Black King's square  
→ Distance  
Q Queening square  
PP Pawn's square

Ex. 2. Rule 29  $Pf=1 \wedge Bf > 3 \wedge (B \rightarrow SD) - (W \rightarrow SD) < -1$  &  $SDR > PPR$   
If TRUE then the position is a DRAW.

Where the symbols are:

Pf Pawn's file  
Bf Black King's file  
SD A square defined by, which breaks down into:  
SDF Square defined by file;  $SDF = 3$   
SDR Square defined by rank;  $SDR = \text{If } WR \leftarrow BR+1 \text{ Then } BR+Bf-3$   
Else BR

PPR Pawn's Effective Rank

Taking the position WP:a3,BK:d4,WKf3, for example, we get:

$$WR \leftarrow BR + 1 \quad \text{so} \quad SDF = 3; \quad SDR = 4 + 3 - 3 = 4$$

Now, the DISTANCE ( $\rightarrow$ ) is evaluated by taking the maximum absolute value of the difference in the respective coordinates of the pieces involved. Hence

$$(B \rightarrow SD) = \text{DIST}(\overbrace{4}^{BK}, \overbrace{4}^{SD}, \overbrace{3}^{SD}, \overbrace{4}^{SD}) = 1$$

$$(W \rightarrow SD) = \text{DIST}(\overbrace{6}^{WK}, \overbrace{3}^{SD}, \overbrace{3}^{SD}, \overbrace{4}^{SD}) = 3$$

$$(B \rightarrow SD) - (W \rightarrow SD) = 1 - 3 = \underline{-2}, \text{ which is } < \underline{-1}$$

and  $SDR = 4$ , which is  $> PPR (3)$ .

So Rule 29 holds true, and the position is a DRAW.

### 5.3.3 Translation of Harris' Program

This program, not surprisingly, proved to be the hardest to translate into an Advice Text. It is long, complex, and procedural. The successful computation of a position value is dependent on the precise calculation and understanding of quite a number of definitions and definitional procedures. Correctly marking the board in order to compute various distance measures is an essential ingredient for the correct use of this Advice Text. Use of scratch paper to store results of computations is also probably essential, since the number of these quickly grows out of hand for the human memory. The necessity for board marking is stressed in the instructions. Throughout the translation process it was difficult to find a balance between maintaining the structure and details of the program while making use of the advantages offered by the English language.

Another point which must be mentioned here is that the Harris Program version being translated was tested as 98.33% correct on Posnums 0-10000, as reported earlier. With changes having been made to Procedures WKOPP and RANK234, the resulting Advice Text was renamed "Harris-Kopec". Thus a small amount of "advice" was being given, which was known to be imprecise or incorrect in the sense that it was known that PROCEDURE RANK234 did not correctly evaluate all the positions it encompassed. However based on the earlier reported empirical testing and on Bitner and Hansche's (1976) comments, it was considered that there was sufficient reason to regard the modified version of PROCEDURE WKOPP (with SSOP and KOPOP) superior to Harris' original version.

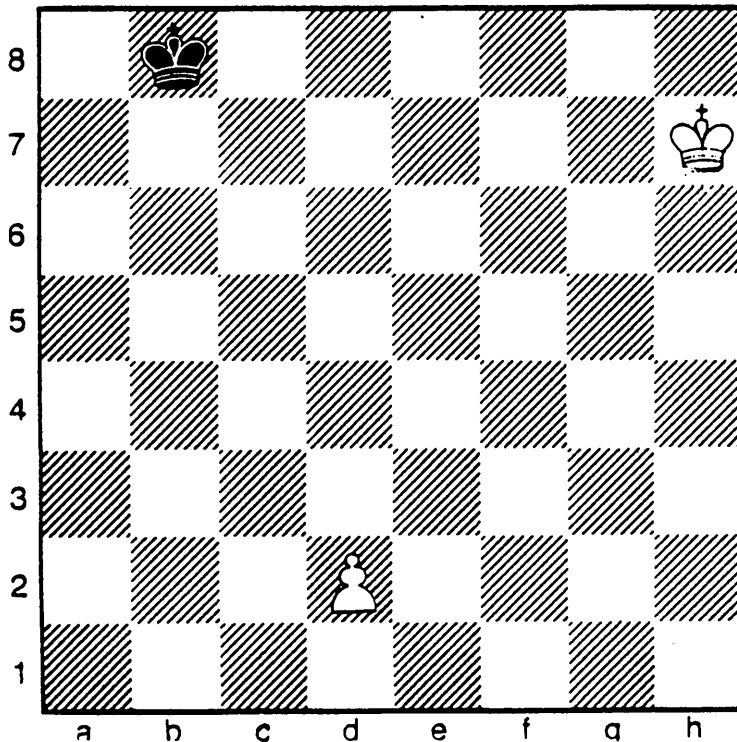
#### 5.3.4 Translation of Bramer's Program

Translation of Bramer's Program into English proved to be a rather straightforward task. Even though this program is comparable to the Harris-Kopec program in length, the structures and computations involved are much simpler, and therefore easier to translate. Most of the 19 classes had already been translated into English (Bramer, 1977), and their concepts were readily understandable by the human chessplayer.

The translation of a number of complex Algol procedures, such as GENMOVES, TIEBREAK, BESTMOVE and ASSFNVAL was greatly curtailed by some careful stepwise instructions, as in Bramer (1980a, p.84), and the use of the table on p.220 of Bramer's Ph.D. thesis (see Section 5.4.3 here, The Bramer Advice Text (page B3)). The instructions were therefore an essential ingredient of the translation process, and these went through a few levels of refinement before being used in the experiment.



5.3.5 Example of Bramer's 19 Equivalence Class Approach for KPK  
 (See Section 5.4.3 for the actual Advice Text.)



For this position let us consider 4 legal successor positions with the move index in Bramer's notation given in "()" : Kg6 (6), Kg7 (7), Kg8 (8), d4 (10).

If we go through the Equivalence Class List for the successor positions resulting from the above moves, we find they all fall into Class 18:

The WKR is greater than the PR (White Wins)

Now turning to the Table on Page B3 we find that Class 18 has Associated Functions 4, 1 and 8 to be applied. Applying these Associated Functions as "tie-breakers" we find that moves (6), (7), and (8) still tie on Associated Function 4: MIN of WK to P file difference,

while move (10) falls out. Hence Associated Function 1 is applied, giving:

MAX of PR

Again moves (6), (7) and (8) tie.

Now we apply Associated Function 8: 8 minus the MIN of WK to P rank difference, giving:

$$\begin{aligned} & 8 - \text{MIN of rank difference of WK to P for moves} \\ & \quad (\text{Kg6, Kg7, Kg8}) \\ & = 8 - \text{MIN}((6-2), (7-2), (8-2)) \\ & = 8 - 4 \end{aligned}$$

Hence the move Kg6 is the only winning move in the position since it is selected from one of the "1733 Critical Positions".

#### 5.4 The Advice Texts

The translation process resulted in three Advice Texts widely varying in size, structure, and complexity. The Beal Advice Text is clearly decision-table-like and straightforward in its application except for a few complex rules and definitions. The Harris-Kopec Advice Text is at the other end of the spectrum, being very procedural and complex in application. The two share the necessity for some intricate definitions. The Bramer Advice Text is in some sense intermediate, being neither decision-table-like nor procedural. Its application may at first appear complex, but once its consistency in form is discovered by the user, he/she will find it rather "friendly". The Bramer definitions are few, and easy to recognize for the chessplayer. We now consider each Advice Text in more detail.

#### 5.4.1 The Beal Advice Text

This Advice Text is given below and consists of 8 pages in all. There is one page for stimulus positions (15) and board representation. All stimulus positions are White-to-move since the Table's 48 Rules are all designed for White-to-move positions. One page (Page C) is devoted to the decision table itself, and one page is for definitions. Of the definitions, SG and SD are rather unpleasant to use. Fortunately these are only used for Rules 28, 29, and 30. The remaining four pages are devoted to the 48 Rules themselves. These for the most part are easy to comprehend, with the exception of Rules 28-31 and 33, which are rather long and cumbersome. For subjects who might not have been acquainted with certain mathematical symbols the definitions of the notation used as given in Table 1 (see Page 10) was provided. Empty boxes in the decision-table mean "don't care" for those parameters involved. It is important to note that if no matching Rule is found, then the position is a draw. In a preliminary version of the Text, the first three of the randomly constructed input positions found no Table matches. It was decided that this was poor design, since subjects might be psychologically affected and thereby lose faith in their answers, or in the Table.

DON BEAL'S PROGRAM AS A KPK ADVICE TEXT

INSTRUCTIONS

On the next page (PAGE B) you will find a number of positions from the ending King and Pawn vs. King (KPK). Use the TABLE on PAGE C to determine which Rule (of 48) matches which position. Definitions are provided on PAGE D. After finding a Rule for a position, determine the "VALUE" (Win or Draw) of each position by looking up the appropriate Rule on the following pages. Enter results on PAGE B in the columns provided.

It is suggested that you try each Rule sequentially. If a Rule has no entry condition, then go to the Rule directly. A Rule is TRUE if all the "AND" conditions hold true.

Scratch paper and chess set/board are provided. Subjects will have one hour to complete the experiment.

Use whatever chess talents or knowledge you have. Don't hesitate to ask questions if you are confused, stuck or need help.

ALL POSITIONS WHITE TO MOVE

	WK	BK	WP	RULE NUMBER	VALUE
1.	d5	d7	b5		
2.	a4	e5	a2		
3.	b2	e7	b3		
4.	d6	d8	d5		
5.	d4	d6	d2		
6.	f3	g6	c3		
7.	h3	f5	d3		
8.	b2	d7	b3		
9.	c5	e6	d3		
10.	h3	h7	c3		
11.	b3	d5	f6		
12.	d5	d8	b6		
13.	d1	f8	c3		
14.	d5	c2	a2		
15.	c6	a8	b5		

	a	b	c	d	e	f	g	h
8								
7								
6								
5								
4								
3								
2								
1								
	1	2	3	4	5	6	7	8

-- FILES --

BOARD REPRESENTATION

ENTRY CONDITIONS

RULE NUMBERS

	PF	PR	WF	WR	BF	BR	WR-PR	BR-PR	WR-BR	BF-PF	WF-PF	BF-PF	BF-WF
1	=1	=7	=1	=8	=3	>6							
2	=1	=6	<4	=6	=3	=8							
3	=1				=1			>0					
4	=1	=7			>2								
5	=1				≤3								
6	=1		=1		=3		=1	=1					
7													
8													
9	=1	≤3	≤2	=8	=4	≥7							
10	=2	=6	≤3	=6	=1	=8							
11	=2	=6	=4	=8	=1	=8							
12		=7		<8									
13		=7		=6							=0		
14		=7		≥6									
15		=6											
16	>1	=6											
17	>1	=6				=8						=1	
18	>1	=6				>6						=2	
19	=1	=6	=1	=8	=2	=6							
20		≥5					=0	=2			=2		=0
21	>1	=5					=1				≤1		
22	>1	=5											
23	>1	=5		≥4				≥2				=3	=0
24	>1	=5											
25		=2				=3						>1	
26							=2	=0			>1	=1	
27	=1		=1		>3				=0				
28									≥-1				
29	=1				>3								
30	=1				>3								
31													
32													
33	>1												
34	>1												
35													
36	>1												
37							=-1	=0			≤2	≠2	
38								=0		>1			
39								=0		<-1			
40	>1							=0				>1	
41	>1							≥3					
42	>1						≥2		<0				
43	>1						≥3		≤1		≤2	≠0	
44	>1						≥0	≥5				≥3	
45	>1	=2				=8							
46												>1	
47								=0		=-2			
48	>2							=0		=2			

NOTE: If no Rule can be found to match a position, then the position is a DRAW

## DEFINITIONS

SG (Goal Square) is comprised of:

SGF : If the WK's file is less than the P's file,  
then the P's file minus 1,  
ELSE  
the P's file plus 1.

SGR : If the WK's file equals the P's file  
and the WK's rank is greater than the BK's rank,  
then the WK's rank minus 1  
ELSE  
the WK's rank minus (the absolute value of the  
difference in files between WK and P) plus 1.

SD (Square Defined) is comprised of:

SDF : = 3

SDR : If (the BK's rank is 1 less than the WK's rank,  
and possibly equal to or greater than the WK's rank)  
then the BK's rank plus BK's file minus 3  
ELSE  
the BK's rank.

T One file towards the pawn. E.g. TBF = If BF > PF then BF-1  
ELSE BF+1.

PPR Square the P is on except when the P's rank equals 2,  
then the square above the P.

Queening Square Square on which P will queen.

distance The distance between two squares, i.e. the maximum  
between (the positive value of the file difference  
between two squares)  
AND  
(the positive value of the rank difference between  
two squares)  
i.e. the shortest number of King moves between two  
squares.



Translation of Beal's 48 Rules

- Rule 1 DRAW
- Rule 2 WHITE WINS
- Rule 3 DRAW
- Rule 4 WHITE WINS
- Rule 5 BK's rank minus PPR rank is greater than 1  
DRAW
- Rule 6 BK's rank is 1 or more greater than the P's  
effective rank (PPR)  
DRAW
- Rule 7 The distance of the BK from the Queening Square is  
greater than the distance of PPR from the Queening Square  
WHITE WINS
- Rule 8 WK's distance from PPR is at least 2 greater than the BK's  
distance from PPR and the BK is not diagonally above the P  
DRAW
- Rule 9 WHITE WINS
- Rule 10 DRAW
- Rule 11 DRAW
- Rule 12 WK is a distance of 2 from PPR and the BK is on the Queening  
Square  
DRAW
- Rule 13 BK is on the Queening Square  
WHITE WINS
- Rule 14 WK's distance from PPR is less than or equal to 2, and the  
BK is not on the Queening Square  
WHITE WINS
- Rule 15 BK's distance from the square 2 ranks ahead and 1 file to the  
right of the P is greater than WK's distance from the square  
to the right of the P,  
AND BK's distance from that square is greater than 1  
WHITE WINS
- Rule 16 BK's distance from the square 2 ranks ahead and 1 file to the  
left of the P is greater than the distance of the WK from the  
square to the left of the P  
WHITE WINS
- Rule 17 WK's distance from the square 2 ranks behind the BK is 1  
WHITE WINS

- Rule 18 WK's distance from the square on the file of the BK,  
rank 5, is less than or equal to 1  
WHITE WINS
- Rule 19 DRAW
- Rule 20 WHITE WINS
- Rule 21 WHITE WINS
- Rule 22 WK's distance from the square diagonally in front to the  
right, above the P, is 1  
AND the BK's distance from that square is greater  
WHITE WINS
- Rule 23 WHITE WINS
- Rule 24 WK's distance from the square diagonally in front to the  
left of the P is 1  
AND BK's distance from that square is greater  
WHITE WINS
- Rule 25 WK's distance from the square 2 ranks above the BK is  
less than or equal to 1  
WHITE WINS
- Rule 26 Kings are on the same side of the P  
DRAW
- Rule 27 WHITE WINS
- Rule 28 BK's distance from PPR minus WK's distance from the SG  
plus the SGR minus the PPR rank is greater than or equal  
to -1  
AND WK's rank is greater than the P's  
AND WK to P rank difference is greater than WK to P file  
difference  
AND BK's distance from SG is greater than WK's distance  
from SG  
WHITE WINS
- Rule 29 BK's distance from SD is two or more less than the WK's  
AND SDR is greater than the PPR  
DRAW
- Rule 30 BK's distance from SD is less than or equal to the WK's  
distance from SD  
AND SDR is greater than the PPR  
AND the file difference between BK and P is less than or  
equal to the file difference between WK and P  
DRAW

- Rule 31 BK's distance from the square diagonally to the right above the P is greater than the WK's distance from that square AND BK's distance from the square 3 above, 1 file to the right of the P is greater than the WK's distance from the square diagonally to the right above P AND the WK is not on the diagonal from the left below the P through the P, to the right above the P  
WHITE WINS
- Rule 32 BK is on the square 3 ranks above, 1 file to the right of the P AND the WK's distance from the square diagonally in front to the right of the P is 1  
WHITE WINS
- Rule 33 BK's distance from the square diagonally to the left above the P is greater than the WK's distance from that square AND BK's distance from the square 3 above, 1 file to the left of the P is greater than the WK's distance from the square diagonally to the left above the P AND the WK is not on the diagonal from the left above the P, through the P, to the right below the P  
WHITE WINS
- Rule 34 BK is on the square 3 ranks above, 1 file to the left of the P AND WK's distance from the square diagonally to the left above the P is 1  
WHITE WINS
- Rule 35 BK's distance from the square 2 ranks above, 1 file to the right of the P is greater than the WK's distance from that square  
WHITE WINS
- Rule 36 BK's distance from the square 2 ranks above, 1 file to the left of the P is greater than the WK's distance from that square  
WHITE WINS
- Rule 37  
WHITE WINS
- Rule 38 WK's distance from the square 2 ranks above, 1 file to the left of the BK is less than or equal to 1  
WHITE WINS
- Rule 39 WK is on or adjacent to the square 2 ranks above, 1 file to the right of the BK  
WHITE WINS
- Rule 40 WK's distance from the square behind the P is less than or equal to 1  
WHITE WINS
- Rule 41 WK is adjacent to the square 2 ranks below the BK  
WHITE WINS
- Rule 42 File difference between BK and P is greater than or equal to the file difference between WK and P  
WHITE WINS
- Rule 43  
WHITE WINS

- Rule 44 The PPR equals 3, file difference between BK and P is 1 less than, equal to, or greater than the file difference between WK and P  
WHITE WINS
- Rule 45 File difference between BK and P is 1 less, equal to, or greater than the file difference between WK and P  
WHITE WINS
- Rule 46 BK's rank equals the PPR  
AND the WK is on or adjacent to the TBF (square one file towards the P from BK) with rank 2 above the BK  
WHITE WINS
- Rule 47 WK is on or adjacent to the square 2 files to the right,  
1 rank below the P  
WHITE WINS
- Rule 48 WK is on or adjacent to the square 2 files to the left,  
1 rank below the P  
WHITE WINS

#### 5.4.2 The Harris-Kopec Advice Text

This Advice Text is given below and consists of 12 pages. One is devoted to instructions; one for stimulus positions (10) and board representation; one for top level procedural flow; two for definitions; five for the actual procedures of the Advice Text, and two final pages exemplifying the board marking necessary for proper use of the Advice Text. The Procedural Flow Diagram (Bitner & Hansche, 1976) is intended to guide the user towards the appropriate Procedures to be used in order to reach a decision on the value (win or draw) of a stimulus position. The two pages of definitions are of two types, one page being for specific board relationships such as AHEAD1, KNIGHTUP, PRANK etc., the other being used to compute distance relationships between two pieces, or a piece and surrounding squares. The Procedures PAWNCANNOTQUEEN and KINGCANNOTTAKEPAWN are intended to determine whether what their names mean is true. The steps involved for both these Procedures are numerous and involved. The human chessplayer will most probably find it easier to decide for himself whether these are true. Finally, there are Procedures RANK234, RANK5, RANK6, and RANK7 which classify a position according to the P's rank. The distance computations involved in these Procedures are complex and their purpose difficult for the human to comprehend. The names given to groups of squares under consideration, which have been defined under "Definitions" should be helpful. Finally, the reasons attached to a value do help in giving some idea of why that result has been obtained.

THE HARRIS-KOPEC PROGRAM AS A KPK ADVICE TEXT

INSTRUCTIONS

On PAGE H2 you will find a number of King and Pawn vs. King (KPK) positions with White and Black to move. Use the Procedural Flow diagram on PAGE H3 to determine the "VALUE" (Win or Draw) of each position by carrying out the Procedures defined on PAGES H6-H10.

Indicate which Procedure allowed you to reach a decision and the reasons for a VALUE (if given, the reason will appear in parenthesis after results).

Definitions which are used in various Procedures appear on PAGES H4 and H5. Definitions specific to a Procedure appear at the top of it. A sample board marking appears on PAGE H11. Scratch paper and chess set/board are provided. It is strongly suggested that subjects try marking the board initially.

Subjects will have one hour to complete the experiment. Use whatever chess knowledge you have to facilitate your understanding of the Procedures. Don't hesitate to ask questions if you are confused, stuck, or need help.

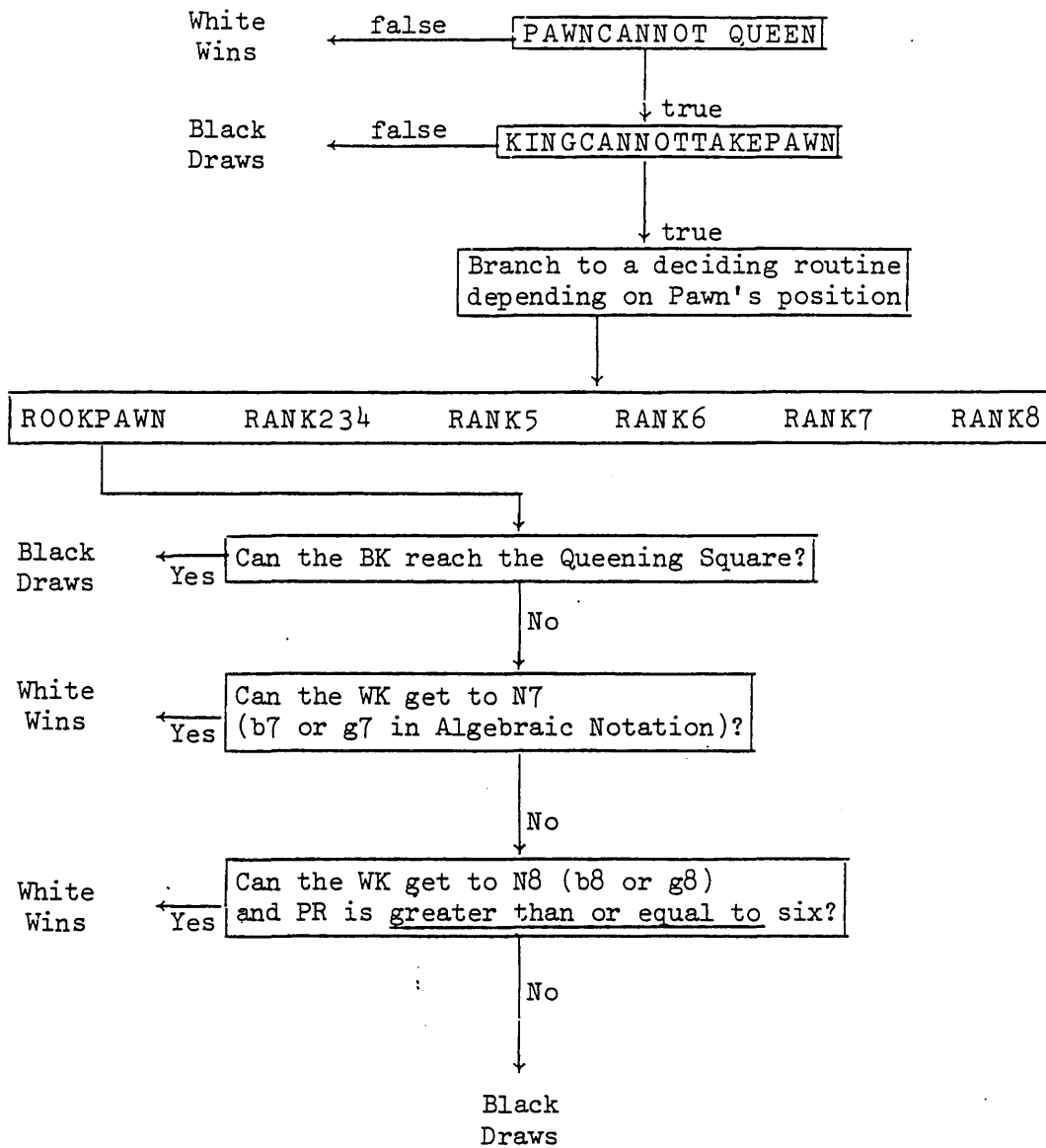
	WK	BK	WP	Side to move	Value	Reason
1.	b2	c7	b3	W		
2.	h3	h7	d3	B		
3.	d3	d6	d4	B		
4.	d5	c2	a2	B		
5.	d5	d7	b5	W		
6.	a5	c5	a2	W		
7.	a4	c6	b3	B		
8.	d1	f8	c3	W		
9.	c3	c7	d3	B		
10.	c6	b8	b5	W		

R A N K S	8	7	17	27	37	47	57	67	77
	7	8	16	26	36	46	56	66	76
	6	5	15	25	35	45	55	65	75
	5	4	14	24	34	44	54	64	74
	4	3	13	23	33	43	53	63	73
	3	2	12	22	32	42	52	62	72
	2	1	11	21	31	41	51	61	71
	1	0	10	20	30	40	50	60	70
		a	b	c	d	e	f	g	h

--- FILES ---

BOARD REPRESENTATION

Procedural Flow of Harris's KPK Program  
 (As in Bitner & Hansche, 1976)





DEFINITIONS

PGUARDLEFT: The square diagonally to the left above the P  
 PGUARDRIGHT: The square diagonally to the right above the P  
 NEVER: A square which can never be reached with regard to a specific K (marked 99)  
 UP: The square directly in front of a P  
 DOWN: The square directly behind a P  
 LEFT: The square to the left of a P  
 RIGHT: The square to the right of a P  
 RANKLEN: The length of a rank (8)  
 INFRONT: The 6 squares in front of a P, i.e.
 

```

X X X
X X X
P

```

INFRONTLEN: Equals 6  
 FLANKLEN: The two square diagonally behind a P, i.e.
 

```

P
X X

```

AHEADOREVEP: The 5 squares ahead or even with a P, i.e.
 

```

X X
X X X
X P X

```

AOEWPLEN: Equals 5  
 AHEAD1: The 3 squares in front of a P, i.e.
 

```

X X X
P

```

ALLEN: Equals 3  
 ASIDE1OR2: The squares 2 to the left and 2 to the right of a P AND the squares immediately to the left and right of a P, i.e.
 

```

X X P X X

```

ASIDE1: The squares immediately to the left and right of a P, i.e.
 

```

X P X

```

ASIDE2: The squares 2 to the left and 2 to the right of a P  
 KNIGHTUP: The 2 squares a Knight jump in front of a P, i.e.
 

```

S X S
X
P

```

 where "S's" are these squares.

RANKSIX: The 6th rank  
 WMOVE: TRUE if W to move, otherwise FALSE  
 BMOVE: TRUE if B to move, otherwise FALSE  
 PFILE: The file of the P  
 PRANK: The rank of the P  
 WKFILE: The file of the WK  
 WKRANK: The rank of the WK  
 BKFILE: The file of the BK  
 BKRANK: The rank of the BK  
 QS: The square on which the P will queen  
 RP: The Rookpawn  
 AHEAD2: The 3 squares 2 ranks ahead of the P, i.e.
 

```

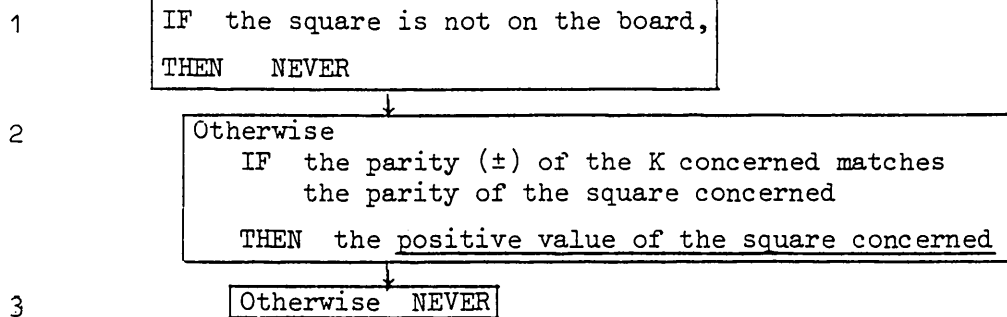
S S S
X
P (marked with "S")

```

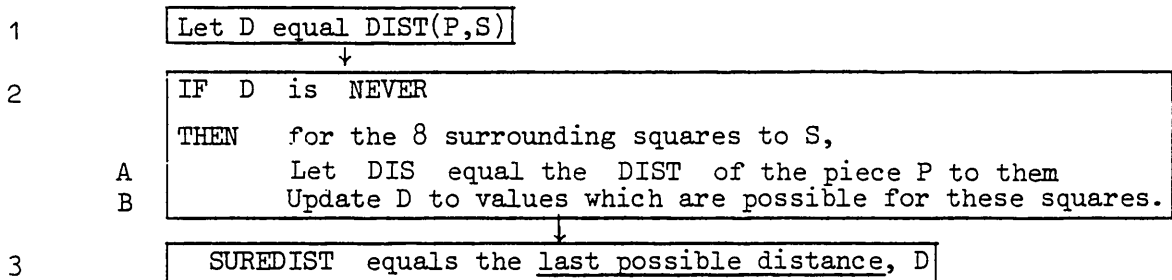
AHEAD2LEN: Equals 3

DEFINITIONS OF DISTANCE PROCEDURES

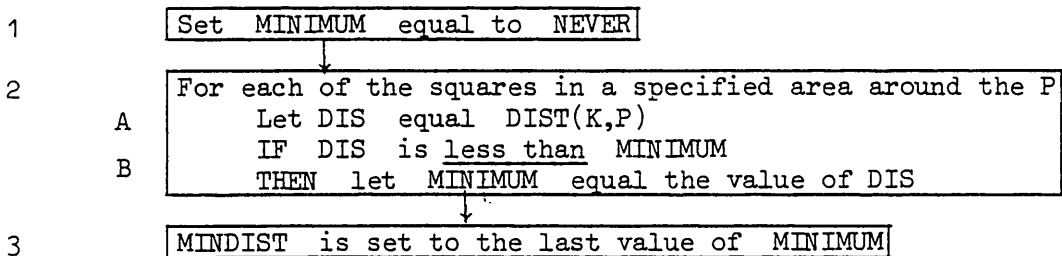
DIST(K,P): The accessibility of a square to a King.



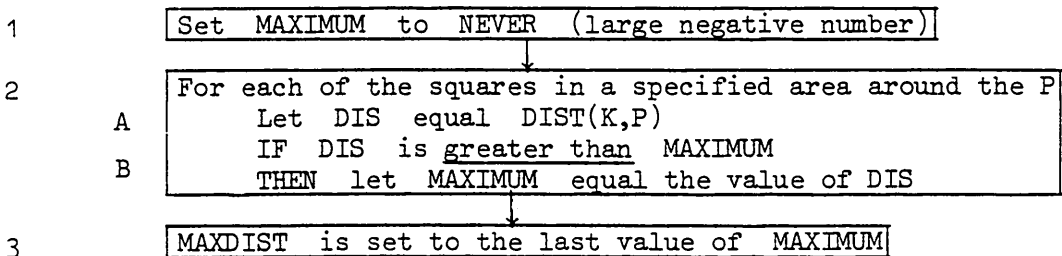
SUREDIST(P,S): The accessibility of a square's immediate neighbours (if the square itself is inaccessible) to a King.



MINDIST(K,OFFSETS,OFFLEN): The minimum distance of a K to a square or group of squares..



MAXDIST(K,OFFSETS,OFFLEN): The maximum distance of a K to a square or group of squares.

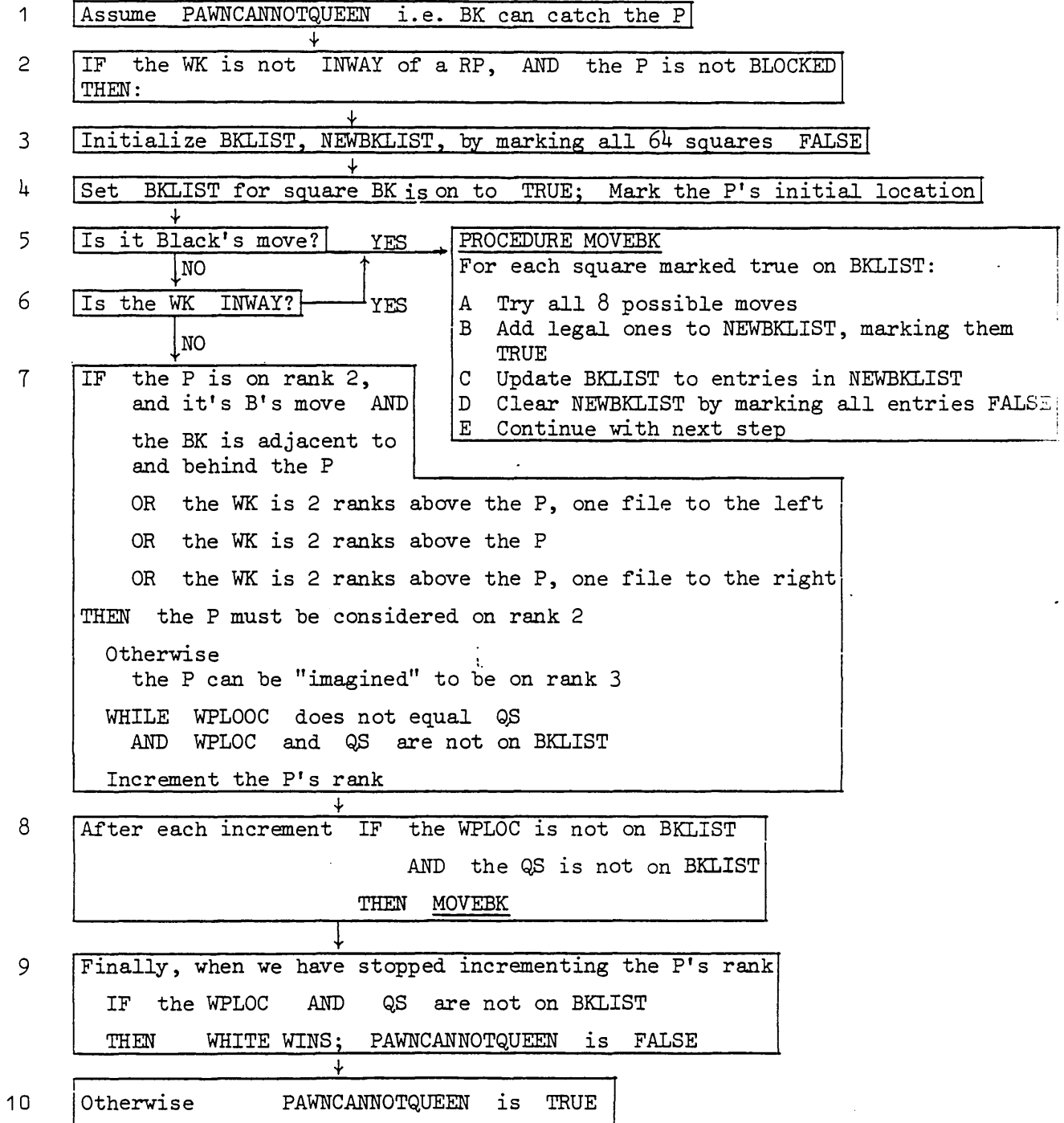


ADJACENT(P,Q): TRUE if squares P and Q are adjacent.

PROCEDURE PAWNCANNOTQUEEN

## DEFINITIONS:

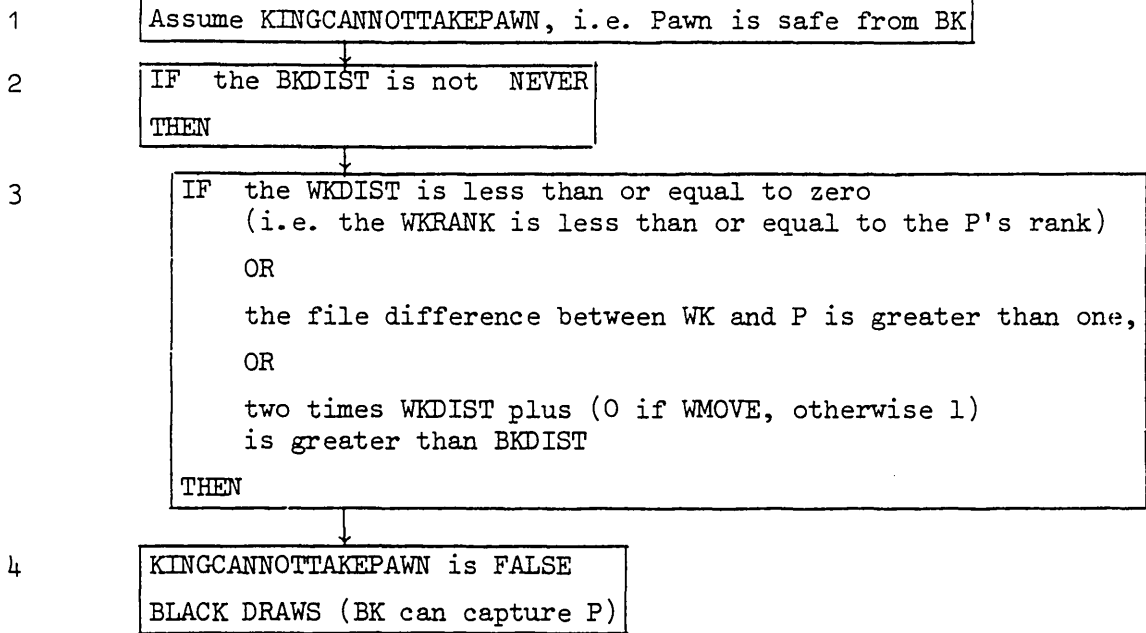
- INWAY: WK is on the file of the P and somewhere ahead of it.  
 BLOCKED: BK is on the file of the P and somewhere ahead of it.  
 WPLOC: Square the P is located on.  
 BKLIST: List of all the squares the BK can reach.  
 NEWBKLIST: List of all the target squares for the BK which are generated by trying legal moves.



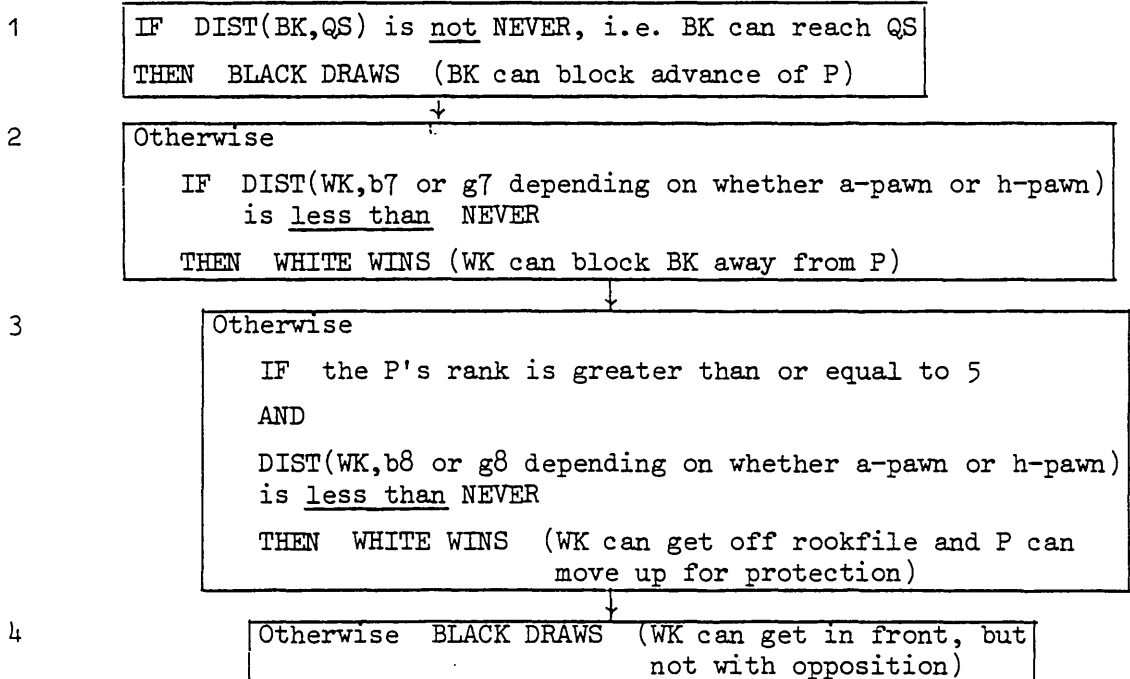
PROCEDURE KINGCANNOTTAKEPAWN

DEFINITIONS

BKDIST equals DIST(BK,P)  
 WKDIST equals WKRANK minus P's rank

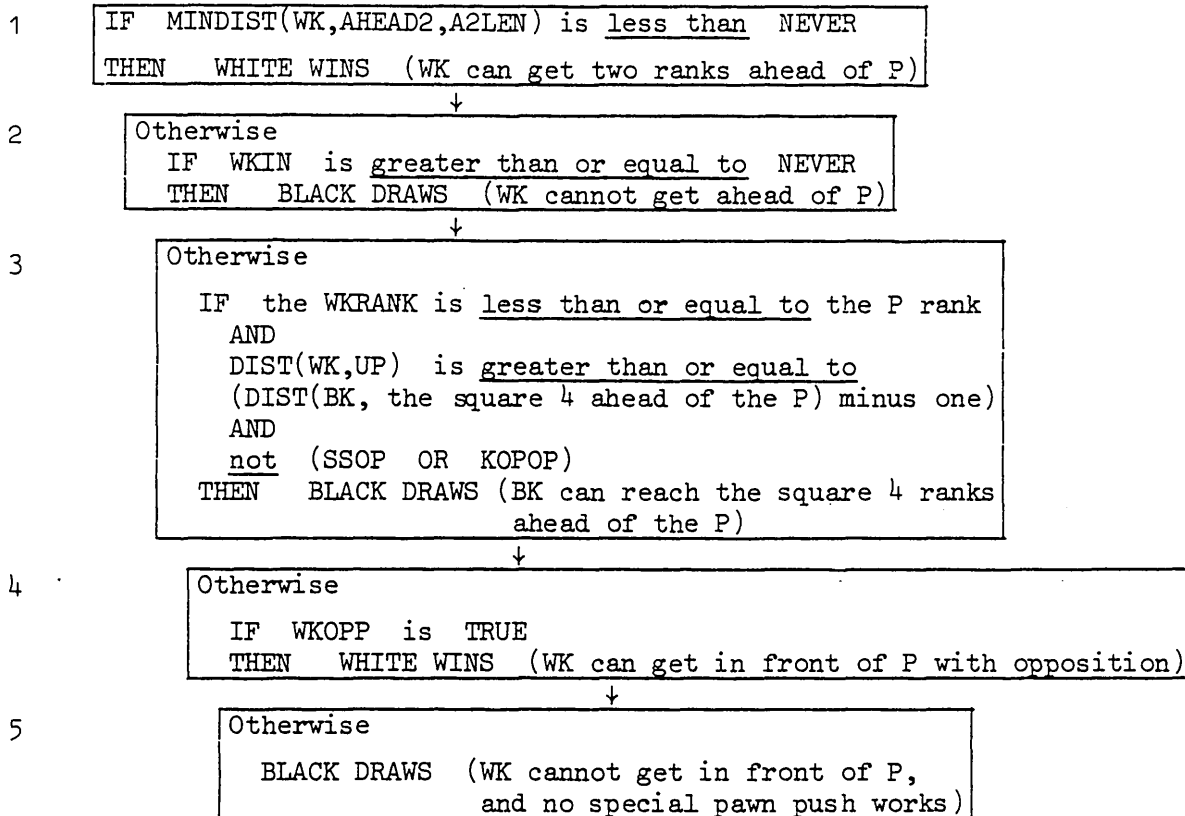


PROCEDURE ROOKPAWN



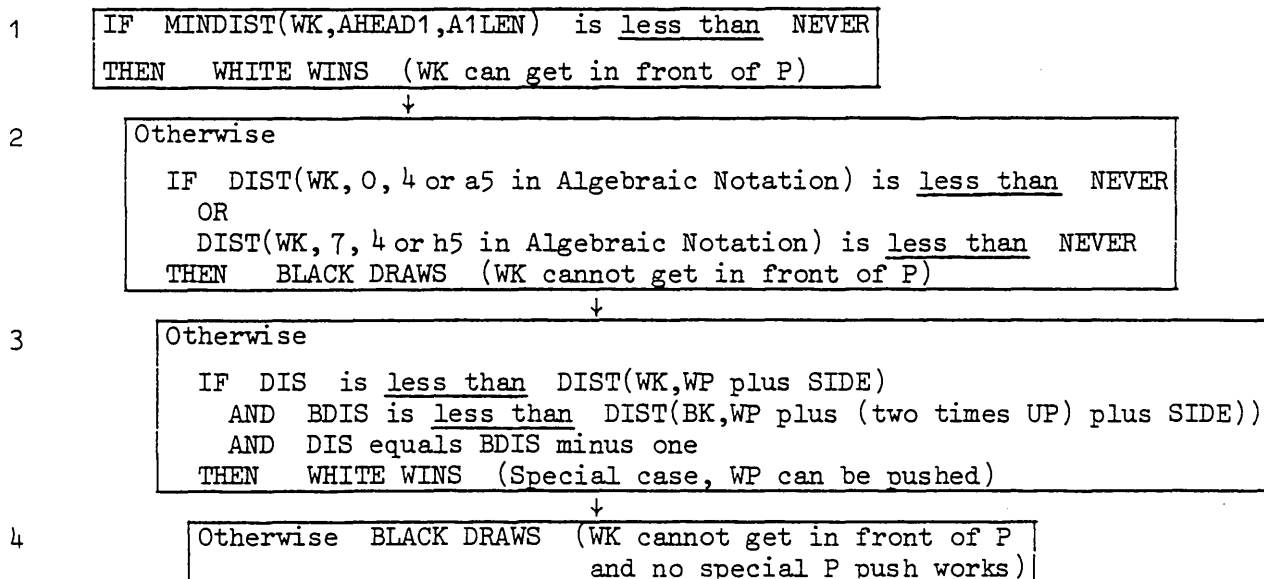
## PROCEDURE RANK234

DEFINITIONS: WKIN equals MINDIST(WK,AHEAD1,3)



## PROCEDURE RANK5

DEFINITIONS: SIDE: IF WKFILE is less than PFILE THEN minus RIGHT  
Otherwise RIGHT  
DIS: DIST(WK,WP plus (two times SIDE))  
BDIS: DIST(BK,WP plus (two times (Up plus SIDE)))



PROCEDURE RANK6

1 IF MINDIST(WK,AHEAD1,A1LEN) is less than NEVER  
 THEN WHITE WINS (WK can support advance of P)

2 Otherwise  
 IF the positive value of the file difference between  
 BK and P is greater than one  
 AND the K's are on the same side of the P  
 AND the WK's rank is less than 5  
 THEN

2 A IF SUREDIST(WK,P plus DOWN) is less than or equal to  
 DIST(BK,WP plus (twotimes UP)) plus one  
 AND the positive file difference between BK and P  
 B is greater than the file difference between  
 (WK and P minus one if WMOVE is TRUE)  
 THEN WHITE WINS (WK can use P to gain opposition)

4 Otherwise  
 BLACK DRAWS (BK can get in front of P before WK)

5 Otherwise  
 IF MINDIST(WK,ASIDE1,2) equals (MINDIST(BK,KNIGHTUP,2)  
 plus one)  
 THEN WHITE WINS (WK can get even with P with opposition)

6 Otherwise.  
 BLACK DRAWS (WK can get even with P, but  
 not with opposition)

PROCEDURE RANK7

1 IF MINDIST(WK,AHEADOREVEP,5) is less than NEVER  
 THEN WHITE WINS (WK can get even with or ahead of P)

2 Otherwise  
 IF DIST(BK,QS) minus MINDIST(WK,FLANK,2) equals -1  
 THEN WHITE WINS (WK can get even with or in front of P  
 with opposition)

3 Otherwise  
 BLACK DRAWS (WK cannot get even with or in front of P  
 with opposition)

PROCEDURE RANK8

WHITE WINS (P has already Queened!)

KING OPPOSITIONPROCEDURE SSOP (Same Side Opposition)

DEFINITIONS: KFDIST equals WKFILE minus BKFILE; ABSKFDIST equals positive value of KFDIST.

1 A IF the positive value of the WK to P file difference is less than  
 B the positive value of the BK to P file difference  
 C AND ABSKFDIST is not one AND it's not BMOVE  
 AND IF BKFILE is less than PFILE  
 THEN DIST(WK,WP plus PGUARDRIGHT) is less than NEVER

2 Otherwise  
 IF DIST(WK,WP plus PGUARDLEFT) is less than NEVER  
 THEN SSOP is TRUE (Tests for case where WK and BK are on the same side  
 of the P and WK can reach a key square on the other side of the P  
 ahead of BK)

PROCEDURE KOPOP (Opposite Side King Opposition)

1 A IF the K's are on opposite sides of the P  
 AND DIST(WK,P plus PGUARDLEFT) is less than  
 DIST(BK,square three ranks ahead, one file to left of P)  
 B OR DIST(WK,P plus PGUARDRIGHT) is less than  
 C DIST(BK,square three ranks ahead, one file to right of P)  
 THEN KOPOP is TRUE (K's on opposite sides of P, and WK can  
 reach Key Square ahead of BK)

PROCEDURE WKOPP (Encompasses All Forms of Opposition)

DEFINITIONS: KFDIST equals WKFILE minus BKFILE; KRDIST equals WKRANK minus BKRANK;  
 ABSKFDIST equals positive value of KFDIST

1 A IF the KFDIST equals zero  
 B OR ABSKFDIST is even and ABSKFDIST equals the positive value of KRDIST  
 C AND (BMOVE is TRUE and KRDIST is even) OR (WMOVE and KRDIST is odd)  
 THEN WKOPP is TRUE

2 A Otherwise  
 B IF the BKFILE equals the PFILE  
 C AND BK is not on the square four ranks ahead of the P  
 AND WMOVE and the WK can immediately reach the square in front  
 of the P  
 THEN WKOPP is TRUE

3 Otherwise  
 IF WMOVE and the WK can immediately land on the square  
 two ranks below the BK  
 THEN WKOPP is TRUE (Standard forms of Opposition can be achieved)

4 Otherwise  
 WKOPP is FALSE

5 IF SSOP OR KOPOP OR WKOPP are TRUE  
 THEN KING OPPOSITION is TRUE

BOARD MARKING

IF WMOVE is TRUE

THEN mark the square of the WK with "-1" and the BK's square "2"

Otherwise

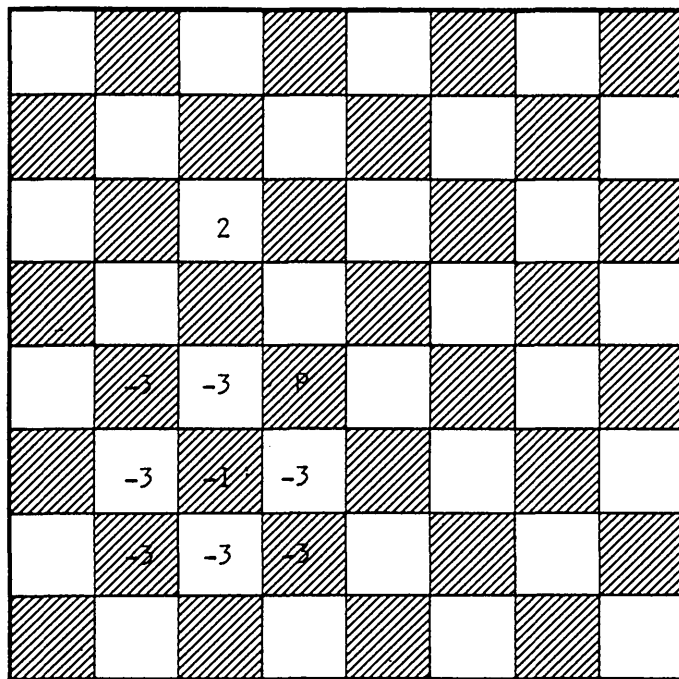
Mark the BK's square with "-1" and the WK's square "2"

Mark the P's square in either of the above two cases with "P"

Start building "HALOS", marking the squares with an increment of 2 by the same sign (+ or -) around each K, alternating for each side to move.

Mark squares neither K can reach with "99".

i.e. if the position is: WK:c3,BK:c6,WP:d4, White to move;



See next page



	4	4	4				
	4	2	4				
99	99	99	99				
	-3	-3	8				
	-3	-1	-3				
	-3	-3	-3				

We continue in this manner until the board is finally marked

6	6	6	6	6	8	10	12
6	4	4	4	6	8	10	12
6	4	2	4	6	8	10	12
99	99	99	99	99	99	99	99
-5	-3	-3	8	-5	-7	-9	-11
-5	-3	-1	-3	-5	-7	-9	-11
-5	-3	-3	-3	-5	-7	-9	-11
-5	-5	-5	-5	-5	-7	-9	-11

FINAL BOARD MARKING

### 5.4.3 The Bramer Advice Text

At the time the production of this advice text was done (1978-79) Bramer's (1981) 20-class KPK program was not yet available. The final form of this Advice Text is given below and occupies 8 pages. One page for instructions, one page again for the 10 stimulus positions (all White-to-play has only 1 winning move), and one page for board representation, which is in algebraic notation; one page is devoted to the table containing the 19 Classes, the Class Values, and corresponding Associated Functions to be applied as tie-breakers in the event 2 or more moves fall into the same Class. The 19 Classes and Associated Functions themselves occupy only four pages. The 10 possible types of moves for White in KPK are defined at the bottom of this page. This was a likely design error in the Advice Text, since the indices for move types could easily be confused with the 9 different Associated Functions. For this reason very precise instructions, and adherence to them, are required for success with this Advice Text. There are one and a half pages of definitions. Again, most chessplayers can surmise the meaning of Procedures STALEMATE and CANRUN for themselves. As mentioned earlier, the 19 Classes themselves should be easy to understand.

THE BRAMER "19 CLASS" PROGRAM AS A KPK ADVICE TEXT

INSTRUCTIONS

On PAGE B2 you will find 10 King and Pawn vs. King (KPK) positions with White to move. Use the Table on PAGE B3 to help you determine the BEST MOVE. Classes and Associated Functions are defined on PAGES B6-B8. Definitions which may be of use appear on PAGES B4 and B5.

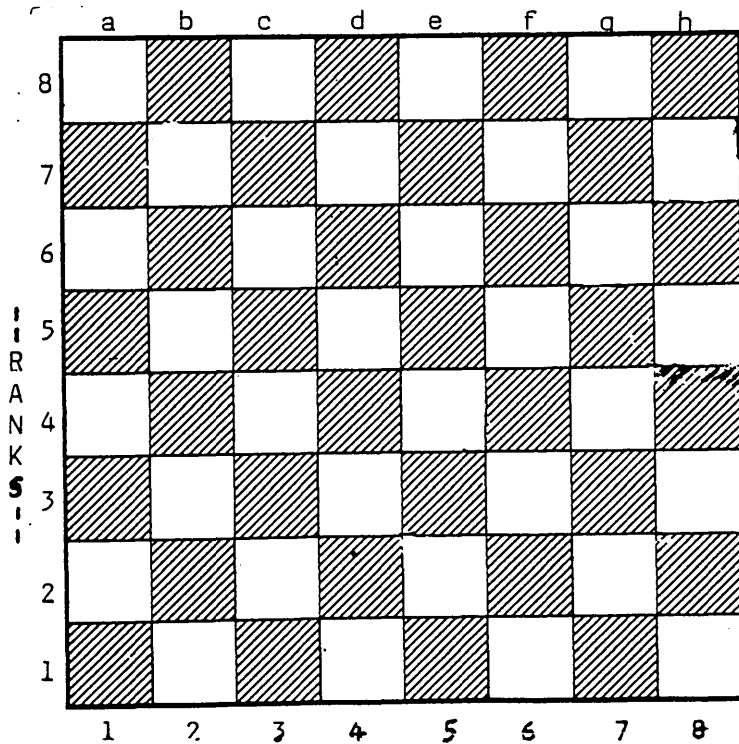
Steps to do the above are:

- (1) Try each legal K move, and then P move of the possible ones (PAGE B3).
- (2) Decide on the move(s) which falls into the highest CLASS VALUE (PAGE B3).
  - a. If 2 or more moves fall into the same Class, ties can be broken by applying the appropriate Associated Functions.
  - b. If there is still a tie, choose the lowest numbered move (PAGE B3).

Scratch paper and chess set/board are provided for any calculations you may need to perform. Fill in your decisions on PAGE B2.

Subjects will have one hour to complete the experiment. Use whatever chess knowledge you have to facilitate your decisions. Don't hesitate to ask questions if you are confused, stuck, or need help.

BRAMER TEST POSITIONS (All Positions White to Move)			BEST <u>MOVE</u>	HIGHEST <u>CLASS</u>	ASSOCIATED <u>FUNCTIONS</u>
WK	BK	WP			
1.	d1	f8	c3		
2.	d5	d7	b5		
3.	c6	b8	b5		
4.	c3	a7	c4		
5.	h7	b8	d2		
6.	c3	b5	d3		
7.	b5	c8	d4		
8.	c3	b5	d5		
9.	g4	b8	d6		
10.	g5	f8	c5		



-- EILES --

THE 19 CLASSES, THEIR VALUE, AND ASSOCIATED FUNCTIONS

CLASS (in order of testing)	CLASS VALUE	ASSOCIATED FUNCTIONS
1	10	2,3,0,0
2	20	0,0,0,0
3	150	0,0,0,0
4	140	1,0,0,0
5	30	2,3,0,0
16	31	0,0,0,0
6	40	1,2,3,0
7	50	0,0,0,0
8	130	1,4,7,0
9	120	1,0,0,0
10	110	1,0,0,0
11	100	1,0,0,0
12	90	0,0,0,0
13	80	1,0,0,0
14	70	1,6,5,0
18	65	4,1,8,0
19	64	0,0,0,0
20	85	0,0,0,0
15	60	2,3,7,9

NOTE: ALL CLASSES ARE BLACK-TO-MOVE POSITIONS



The 8 possible types of K moves.

A P on Rank 2 can make a "9" or a "10" move. Ordinary P's make a "9" move.

PAGE B4  
DEFINITIONS

PROCEDURE STALEMATE

(Boolean) i.e. TRUE or FALSE  
S, S1, S2 are integers;  
If the BK's rank is 8 AND (the WK's file equals the  
BK's file) AND (the WK's file equals the P's file)  
AND (the WK's rank is 6) AND (P's rank is 7)  
THEN TRUE  
If the BK's file equals 1 THEN set S to 1  
If the BK's file equals 8 THEN set S to -1  
Otherwise FALSE  
Set S1 to BK's file plus S; Set S2 to BK's file plus  
(2 times S); (If the WK's file equals S2) AND (the  
WK's rank is greater than 6) AND (the P's file equals S1)  
AND (the P's rank equals 6)  
OR  
(If the WK's rank equals 6) AND (the P's rank equals 7)  
AND ((If the WK's file equals the BK's file) AND (the  
P's file equals S2) OR  
(the WK's file equals S1) AND (the P's file equals S2))  
THEN TRUE  
otherwise FALSE

PROCEDURE DIST (A,B,C,D):

The Maximum of the Absolute (Positive) Value of the differences  
(A-C), (B-D); i.e. the number of K moves between 2 squares of  
file and rank A, B and C, D respectively.

PROCEDURE CANRUN: (Boolean)

IF The BKR is at least 2 less than the PR  
OR  
the BK's file distance from the P minus 1 is greater than  
the effective number of moves till the P Queens  
OR  
the WK to P file distance is 1, the PR is 5, and WKR is 7  
OR  
the WK to P file distance is 1, the PR is greater than 5,  
and the WKR is greater than 6  
OR  
the PR is 7, the WKR is 6, and the WK to P file distance  
is 2 or 1  
AND  
the WKF equals the PF OR (the WKF is less than the PF  
AND the BKF is less than the PF) OR (the WKF is greater  
than the PF AND the BKF is greater than the PF)  
THEN TRUE  
Otherwise FALSE

BKINC: IF the Absolute Value of (the BK to P file difference)  
equals the BK to P rank difference  
AND the BK's rank is at least 2 greater than the P's  
THEN BKINC equals 1  
Otherwise BKINC equals ZERO

RPAWN: IF the P's file equals 1  
THEN TRUE  
Otherwise FALSE

PPR: IF PR equals 2  
THEN 3  
Otherwise PR

WKF: The WK's file

WKR: The WK's rank

BKF: The BK's file

PAGE B6  
TRANSLATION OF BRAMER'S 19 CLASSES  
INTO ENGLISH

NOTE:

All Classes are defined as Black-to-Move.

- Class 1 The P can immediately be captured (DRAW)
- Class 2 The BK is in STALEMATE (DRAW)
- Class 3 The P is on the 8th rank (WIN)
- Class 4 The P CANRUN (WIN)
- Class 5 The BK's distance from the P, plus BKINC is less than the WK's distance from the P (DRAW)
- Class 16 RPAWN AND  
 WKF equals the PF and the WKR is greater than the PR and the BK is adjacent to the square 2 files to the right of the WK  
 OR  
 The distance of the BK from the square 3,8 is less than the distance of the WK from that square  
 OR  
 The WKF is equal to the P file and the WK'S rank is 2 or more greater than the PR and the BK is adjacent to the square on file 3, 1 rank below the WK  
 OR  
 The PR is greater than 2, WKR is 4 greater than the P's, and the BK is adjacent to the square on file 3, 2 ranks above the P  
 OR  
 WKF equals PF, WKR is 3 greater than the P's, and the BK is adjacent to the square on file 3, 2 ranks below the WK (DRAW)
- Class 6 The BK can immediately occupy the square in front of the P (DRAW)
- Class 7 The WK is on the square in front of the P and B can take the opposition (DRAW)
- Class 8 The WK is 2 or more files closer to the P than the BK and not below the rank of the P (WHITE WINS)
- Class 9 The WKR equals the BKR AND the WKR is 1 or 2 greater than the PR AND the file difference between the WK and BK is 2, and between K's and P is 1 (WHITE WINS)



- Class 10 The WKR is greater than the PR AND the WKR minus the PR is less than or equal to the PPR AND the K's are on files adjacent to the P AND the BKR is 1 greater than the WKR (WHITE WINS)
- Class 11 The WK is somewhere in front of the P, on the same file (WHITE WINS)
- Class 12 The Kings are in opposition (i.e. the WKF equals the BKF and the BKR is 2 greater than the WKR) and the WKR is greater than the PR AND The WKR minus the PR is less than or equal to the PPR AND the WKF is adjacent to the PF (WHITE WINS)
- Class 13 The K's are in opposition and the WKR is greater than the P's (WHITE WINS)  
OR  
The K's are in opposition and the WKR equals the PR (DRAW)
- Class 14 The WK is 1 or 0 files from the P AND the WKR is greater than the PR AND the WKR minus the PR is less than or equal to the PPR (WHITE WINS)
- Class 18 The WKR is greater than the PR (WHITE WINS)
- Class 19 The P's rank equals 6, the WKR equals 5, the BKR equals 7 and the WKF equals the BKF (opposition), and the file difference between the WK and the P is 2 (WHITE WINS)
- Class 20 The WKR is equal to the PR, which is 5, and the BKR equals 7 AND the file difference between WK and P is 2, and the file difference between BK and P is 3 AND BETWEEN is TRUE (WHITE WINS)
- Class 15 (Residual Class) All positions so far unclassified fall into this Class.

ASSOCIATED FUNCTIONS

MAX:

Stands for Maximum; MAX of the values computed is to be taken in each case.

<u>Number</u>	<u>Definition</u>
1	MAX of PR
2	8 minus the MAX of the file difference and the rank difference between WK and P
3	8 minus the MIN of the file difference and the rank difference between WK and P
4	MIN of WK to P file difference
5	MAX of WK to P file difference
6	MAX of WK to P rank difference
7	MAX of WR
8	8 minus the MIN of WK to P rank difference
9	MAX of WK to BK file difference

#### 5.4.4 Bramer's KPK Program as a Human Window Example

Comprehensibility and memorizability go hand in hand. The following is a commentary as seen through the eyes of a chess master.

The value (win or draw) of every legal KPK position with White-to-move can be determined by considering the Class of Black-to-move successor positions. Note that text not in the Classes which follow, the text between the parentheses is to facilitate understanding of the actual Class definition used in the Bramer Advice Text, which sometimes precedes it.

- Class 1     The P can immediately be captured.                     (DRAW)
- Class 2     The BK is in STALEMATE.                                     (DRAW)
- Class 3     The P is on the 8th rank.                                     (WIN)  
(It has already been determined that the P cannot be captured (not Class 1) and there is no Stalemate (not Class 2) so the P can safely be promoted to a Q).
- Class 4     The P CANRUN.   (WIN)  
(This definitional Class is comprised of the cases:  
1) The BK is 2 or more ranks below the P.  
2) The BK is outside the square of the P where in both Class 1 and Class 2 above he cannot catch the P.  
3) The P can simply advance to promotion since the WK can provide protection and the BK cannot immediately capture it.  
4) Same notion as Class 3 above.  
5) The P is on the 7th rank and is sure to queen with help from the WK.)
- Class 5     (All the above Classes are not satisfied and the BK is closer to the P than the WK.)                     (DRAW)

- Class 16 (Rookpawn and one of the following cases is true:  
 1) The WK can be "pinned" onto the R-file by use of horizontal Opposition, i.e. the BK is 2 squares to the right of the WK.  
 2) The BK can reach the key defensive square c8.  
 3) Same notion as Class 1 above.  
 4) The BK is sufficiently close to stop the P or pin the WK to the R-file.  
 5) Same notion as Class 4 above.) (DRAW)
- Class 6 The BK can immediately occupy the square in front of the P. (But cannot capture it (Class 1 above).) (DRAW)
- Class 7 The WK is on the square in front of the P and B can take the Opposition (i.e. the BK can step onto the square on the same file, 2 ranks above the WK). (DRAW)
- Class 8 The WK is 2 or more files closer to the P than the BK and not below the rank of the P. (WHITE WINS)  
 (The WK can "block out" the BK from the P.)
- Class 9 (An "arcade" has been built for the P to walk through to promotion.) (WHITE WINS)
- Class 10 (The WK is a limited number of squares ahead of the P, the K's are on opposite sides of the P on adjacent files, and the WK is assured of the Opposition ahead of the P.) (WHITE WINS)
- Class 11 (The WK is on the files of the P, ahead of it and the BK cannot capture the P or obtain the Opposition.) (WHITE WINS)
- Class 12 (The K's are in Opposition, the WK is a limited number of squares ahead of the P and the WK is on a file adjacent to the P's.) (WHITE WINS)
- Class 13 (The K's are in Opposition and the WK is ahead of the P (WHITE WINS)  
 OR  
 The K's are in Opposition and the WK's rank is equal to the P's.) (DRAW)
- Class 14 (The WK is 1 or zero files from the P and ahead of it to a limited degree.) (WHITE WINS)
- Class 18 (The WK's rank is greater than the P's, assuming the above conditions have not been satisfied.) (WHITE WINS)
- Class 19 The P's rank is 6, the WK's rank is 5, the BK's rank is 7, and the WK's file equals the BK's file (Opposition) with a file difference of 2 between the WK and P. (Special Case) (WHITE WINS)

Class 20 (Assures that Class 19 can be achieved.)  
(WHITE WINS)

Class 15 (Residual Class) (All remaining positions).  
(WHITE WINS)

## 5.5 Results of Early Experiments

The preliminary results were of particular help in finding many ways of improving, correcting and refining the Best Advice Text. The Definitions Page was slightly modified. It was stressed at the bottom of the Decision-Table that if no matching Rule was found, then the position is a Draw. Finally, the instructions for this Advice Text were made more explicit.

In scoring the results of the actual experiment we are concerned with essentially one factor, intelligibility from the human user's point of view. This can be measured by answering two immediately obvious questions:

- 1) Can humans use an advice text for a specialized knowledge domain designed from a computer program in the manner we have described?
- 2) Which of our three specimen Advice Texts is the best "teacher" in terms of being usable by the human chess novice?

To answer the first question properly we are interested in two pieces of information: A) How well would a group of chess players perform in finding the correct value or moves from our standardized set of KPK positions? B) The percentage of correct values for some correct reason(s) from the total number of response-events, where a response-event is the pair of answers consisting of an Advice Text's application (depending on text used, could be "wrong", "right", "partially wrong", or "partially right") and the value or move given for a position.

As we can see from the tabulations of our limited data in Tables 5.3, 5.4, and 5.5, the Advice Texts compare reasonably closely on the percentage of correct or partially correct uses of an Advice Text. These are:

Beal	54.5%
Harris-Kopec	44.4%
Bramer	41.7%

Hence a correct answer from Beal's is slightly more likely to be for some right reason than a correct answer from Harris-Kopec's or Bramer's. We can also see from these tables that the percentages of spurious right answers, i.e. situations where the right value or move for a position has been given, despite incorrect application of the Advice Text, are also similar:

Beal	16.4%
Harris-Kopec	24.4%
Bramer	27.8%

Summing the above percentages for the different types of correct value-responses, we get as total percentages of correct responses:

Beal	70.9%
Harris-Kopec	68.8%
Bramer	69.5%

These percentages are remarkably close and when considered in light of the results with control subjects (see Appendix C) might give the impression that prior exposure to advice texts has no effect on accuracy of evaluation of KPK positions. Subsequent testing on novices gave different results as will be seen in Chapters 6 and 7.

To attempt to answer the second question, we consider the matched-pairs results of administration of two Advice Texts in succession to a given subject. Unfortunately the number of such results is too small to show any significant trends. A few

results can, however, be ascertained (Table 5.6):

- 1) Subjects seemed to be confused if they used the Harris-Kopec after Beal.
- 2) The overall average percentage of correct answers with correct application of an Advice Text as the first of a matched-pair was considerably higher for Beal's than for Harris-Kopec's (58.4% vs. 39.8%).
- 3) The percentage of correct answers with correct application was generally considerably higher for Beal's than for the Advice Text paired with it.

From these data we cannot draw firm conclusions about how the Bramer Advice Text compares with the other two as a "teacher", especially as the first of a matched pair. However, it was consistently observed that when the subject understood and followed the instructions for the Bramer Advice Text he did very well and considered it a learning experience, otherwise he got quite stuck, or demonstrated a complete misunderstanding.



Table 5.3

EXPERIMENTAL RESULTSBEAL ADVICE TEXT

Subjects	Rating	Tabulations						
		RR	RW	W*R	W*W	W/R	W/W	T
1. H. Borland	1625	2	0	1	0	2	3	8
2. H. Urquhart	1535	0	0	0	0	2	2	4
3. D. Ward	1725	6	0	2	0	0	0	8
4. R. Ratcliffe	1365	5	0	2	0	3	5	15
5. T. Lacey	1485	1	0	4	0	2	1	8
6. L. McGregor	1680	5	0	2	0	0	0	7
7. J. Blaikie	1250	0	0	0	0	0	5	5
Totals		19	0	11	0	9	16	55

Average rating 1524

$$\text{Percent correct value with matching} = \frac{RR + W*R}{T} = \frac{30}{55} = 54.5$$

$$\text{Percent correct value with no-matching} = W/T = 9/55 = 16.4$$

KEY

Categories:	RR	RW	W*R	W*W	W/R	W/W
RULE:	RIGHT	RIGHT	WRONG but matching	WRONG but matching	WRONG non- matching	WRONG non- matching
VALUE:	RIGHT	WRONG	RIGHT	WRONG	RIGHT	WRONG

T = Total number of response-events (correct and incorrect)

U = Number of opportunities per subject (15).

Table 5.4

EXPERIMENTAL RESULTS  
HARRIS-KOPEC ADVICE TEXT

Subjects	Rating	YR	YW	NR	NW	T
1. A. Mountford	1625	1	0	2	0	3
2. H. Urquhart	1480	0	0	1	3	4
3. B. Ratcliffe	1365	0	0	2	4	6
4. R. Kelly	1735	2	0	2	1	5
5. N. McGregor	1600*	6	0	1	0	7
6. L. McGregor	1680	7	0	2	1	10
7. J. Blaikie	1250	4	0	1	5	10
Totals		20	0	11	14	45

Average rating 1534

Percent Correct Value with Advice Text correctly applied =  $\frac{YR}{T} = \frac{20}{45} = 44.4$

Percent Correct Value with Advice Text incorrectly applied =  $\frac{NR}{T} = \frac{11}{45} = 24.4$

KEY

	YR	YW	NR	NW
ADVICE TEXT CORRECTLY APPLIED:	YES	YES	NO	NO
VALUE:	Right	Wrong	Right	Wrong

T = Total number of response-events (correct and incorrect)

U = 10

\* Rating is an approximate estimate.

Table 5.5

EXPERIMENTAL RESULTSBRAMER ADVICE TEXT

## Main Tabulations

Subjects	Rating	FR	FW	PR	PW	NR	NW	T
1. D. Ward	1725	3	0	2	2	0	0	7
2. R. Kelly	1735	3	0	3	0	2	1	9
3. T. Lacey	1485	0	0	0	0	3	7	10
4. N. McGregor	1600*	2	0	2	1	5	0	10
Totals		8	0	7	3	10	8	36

Average rating 1635

		FR	FW	PR	PW	NR	NW	T
1. E. Campbell	1580	1	0	3	1	2	0	7
2. F. Taylor	1610	0	0	1	3	0	0	4
3. R. McKay	1300	0	0	0	1	0	2	3
Totals		1	0	4	5	2	2	14

Percentages - Main Tabulations

Percent correct value with advice text partially

$$\text{or fully correctly applied} = \frac{FR + PR}{T} = \frac{15}{36} = 41.7$$

Percent correct value with incorrect advice text

$$\text{incorrectly applied} = \frac{NR}{T} = \frac{10}{36} = 27.8$$

KEY

Categories:	FR	FW	PR	PW	NR	NW
ADVICE TEXT CORRECTLY APPLIED:	YES FULLY	YES FULLY	YES PARTIALLY	YES PARTIALLY	NO	NO
MOVE SELECTED:	RIGHT	WRONG	RIGHT	WRONG	RIGHT	WRONG

T = Total number of response-events (correct or incorrect)

U = 15

\* Rating is an approximate estimate.

Table 5.6

Matched-Pairs Results

<u>Subject</u>	<u>Matched Pairs</u>	<u>Percentages</u>		
		<u>% R(1)</u>	<u>% R(2)</u>	<u>% R(3)</u>
1. H. Borland	(1,2)	37.5	0.0	
2. L. McGregor	(1,2)	100.0	70.0	
3. B. Ratcliffe	(1,2)	33.3	0.0	
4. J. Blaikie	(2,1)	0.0	40.0	
5. H. Urquhart	(2,1)	0.0	0.0	
6. A. Mountford	(2,3)		33.3	Not tab.
7. R. Kelly	(2,3)		40.0	66.6
8. N. McGregor	(2,3)		85.7	40.0
9. T. Lacey	(1,3)	62.5		0.0
10. D. Ward	(3,1)	100.0		71.4

KEY

(1) = BEAL; (2) = HARRIS-KOPEC; (3) = BRAMER.

R = Number of fully correct or partially correct applications of Advice Texts giving correct value or move.

Note: Percentages given are based on total of response-events in each case.

In each case the pair of numbers reading from left to right represents the order in which the two Advice Texts were administered, i.e. (2,1) means that Advice Text 2 was administered, followed by Advice Text 1.

## 5.6 Summary and Conclusions

We have compared four programs which compute essentially the same function, in the same computer language (ALGOL-60), on the same machine (DEC-10). The function computed is the game-theoretic value of all legal 179,656 KFK positions after allowing for right-left symmetry, considering White-to-move and Black-to-move positions separately. Although conventional algorithmic-type solution-programs for the evaluation of the above function can be produced, a different tradeoff between storage-space and execution-time is necessary to bring the given solution-program within the bounds of the human window.

The question of "computational efficiency" (i.e. efficiency being inversely proportional to the reciprocal of the product of processor-time and machine memory) has been considered with regard to our four specimen programs. In this regard we found that the processor-time required to run the Beal program (.06 sec.) was comparable to the time it took to access the Clarke database (.05 sec.), while the Harris and Bramer programs were comparable with each other (.15, .18 secs., respectively). On machine memory (words in core) the Beal program and Clarke database access routine were again comparable (approx. 3600) as were the Harris and Bramer programs (approx. 5600, 7200 words, respectively). However we must not forget that the Clarke database's actual computational efficiency is compounded by a data-file which occupies approximately 82,000 words on disk. Hence the order of computational efficiency, measured from the space-time product (see above), from highest to lowest was Beal, Harris, Bramer, and Clarke.

These four programs perform their tasks correctly in all or nearly all cases. Differences between programs arise in the number, "grain size", and concomitant complexity of their constituent patterns. When translated back into the English language the programs (excluding for the moment Clarke's "database program") are distinguished as:

1) The Harris-Kopec Advice Text, which used 7 overall patterns

(correspondingly a lot of calculation)

2) The Bramer Advice Text, based on 19 derived and extended text-book patterns

(considerably less calculation)

3) The Beal Advice Text, consisting of 48 patterns in the form of a decision table

(very little calculation).

Our hypothesis was that when these translated programs are presented to novice human chess players as open book "cribs" (advice texts), performance will be as follows:

1) The Harris-Kopec program: Subjects can understand, cannot carry out.

2) The Bramer program: Subjects can understand, can carry out.

3) The Beal program: Subjects cannot understand, can carry out.

The statistical results of the tests done using these three advice texts have been too limited to confirm this hy-

pothesis. However, the verbal testimony of subjects placed the Harris and Bramer advice texts in the "can understand" category, in conformity with the foregoing prediction. It remains unclear which advice text required the least of a human subject's brain resources, and therefore was most "executable", although the indications pointed to Beal's. Again this is in line with the "human window" hypothesis. The notion of executability as used here is comprised of four components:

- (1) syntax
- (2) semantics
- (3) time-complexity
- (4) space-complexity

(1) Syntax errors mean that a subject may get stuck at some step within the advice text or end up giving wrong answers because some symbol, word, format or instruction has been misunderstood. In this context we label such misunderstandings as errors in "local understanding", while in general when "understanding" is referred to we are concerned with the "global understanding" of some KPK concept an advice text may be attempting to convey. Other syntax errors may occur due to faults in the design or translation of a program into an advice text. (2) Semantic errors involve the misinterpretation of an instruction in an advice text. This misinterpretation would occur after the specific components of an instruction have been correctly processed. Semantic errors are usually conceptual in nature. These may sometimes be hard to distinguish from syntax errors where the specific components have been incorrectly assembled grammatically and thereby

an instruction cannot be carried out. (3) Time-complexity characterises representations which require too much human computation time to lie within the bounds of the "human window". (4) Space-complexity refers to errors due to too large a short-term memory requirement, as in the deeply nested constructions of the Harris-Kopec Advice Text and to a lesser extent the Beal Advice Text. For any combination of the above reasons a subject may not be able to execute the steps in an advice text to obtain correct answers. In post-mortem questioning Beal subjects plainly stated that despite finding that they could carry out the necessary steps to use this advice text, they had learned little or nothing from it: in the "global" sense they had not understood it. The experiments were not, however, specifically designed to test this point.

Most of Beal's 48 rules define what can be classified as "special cases" and not concepts. The limited number of responses given by Harris-Kopec advice text subjects indicates that, as expected, it was not always executable in real time, although subjects stated that they could clearly see its potential as a learning aid. The data for Bramer subjects was really too limited to reach any conclusions about its value as an advice text.



## VI THE CLARKE ADVICE TEXT

### 6.1. Objectives

Can novice human chessplayers execute and learn from an advice text which details the usage of a database for the lookup of the minimax-optimal values of all the possible 98,304 KPK configurations?

Here we are addressing the issue of the cognitive efficiency of a representation (recall that this was defined as the computational efficiency with respect to the "brain machine") and what such a "brain-oriented" representation might look like? In the context of the human window (an interval on the memory-requirement axis of a plot of execution time versus memory space) a Clarke Advice Text clearly lies in the realm of too extensional, for here the grain-size is very small, and the number of grains, namely the 98,304 possible KPK configurations, is too large to be memorized. In the following experiment, however, this cut-off point boundary for the window was nullified by provision of an external store, namely access to the Clarke database itself. Such a representation is in itself opaque to human intelligibility. The questions we attempt to answer are:

- 1) If provided with the Clarke database as a kind of external memory, can a human chess novice perform the computations necessary to look up a position? In other words

is the advice text executable?

2) As a result of using the Clarke database do the subjects learn concepts relevant to the mastery of the KPK endgame?

The distinction here between "concept learning" and "Skinner-type learning" (i.e. based on a stored rote-dictionary of stimulus-response entries) is crucial and most difficult to demonstrate. In the main, we consider the issue whether any form of learning or understanding is possible.

When considering the question of executability, there are four types of errors which were discussed in the previous section:

- (1) syntax
- (2) semantics
- (3) time-complexity
- (4) space-complexity

(1) Syntax errors here would involve a subject's inability to carry out some step in the advice text due to the inability to interpret some symbol, word, format or instruction. (2) Semantic errors in this case could involve such problems as misunderstandings in the rules of KPK, the notion of White's "winning" and Black's "drawing", misunderstanding of the meaning of various symbols in the database, and the method for decoding their value.

(3) Time-complexity errors refer to errors due to the requirement of too much computation for the human. (4) Space-complexity errors have already been referred to in the requirement of too large a short-term memory for the human. Since by executing the steps in the Clarke Advice Text in order to obtain a correct answer, subjects were not required to memorize anything, space complexity errors were not an issue, except to the extent that too large a requirement of such external memory might cause difficulties of accurate retrieval.

## 6.2 Experimental Design

Amongst schoolchildren 15-17 years of age, two groups of about 10 subjects each were sought: chess novices (N) who knew the legal moves and rules, and those who did not know the legal moves and rules (NK). In all cases a WK, BK, WP and a chess board were provided and subjects were invited to use them throughout the experiment. The experimental steps were: (I) The group N was given a "Pre-Test" on 10 critical positions (KPK positions where White to play, has only one winning move) to find out what prior knowledge these subjects had. (Ia) The NK group of subjects were then taught the rules of KPK by a "KPK Manual" and tutorial coaching if necessary. (II) Subjects in each group were then taken by hand through a number of worked examples on the use of the Clarke Advice Text. The time and number of examples required (up to 5 worked examples had been prepared) was

recorded until both the subject and experimenter were satisfied that the subject had been able to carry out the required task for obtaining a correct answer from the database. (III) All subjects were then given a work sheet on which were provided 10 KPK positions and their appropriate Position Numbers for which they were to indicate their opinions for the game-theoretic value with White and Black to move. These were then to be compared with the looked-up minimax-optimal game-theoretic values which were decoded from a listing of the KPK database. (IV) Finally all subjects were given a "Post-Mortem Test" on 10 new positions each of which were specifically related to those in part III in that they were translations or shifts left, right, up or down. Of these 10 White to move positions, 6 were critical, 3 had more than one winning move, and one was a draw.

Thus after presenting NK subjects with our KPK manual and asking if they had any queries, we assumed that such subjects knew the legal moves and objectives for KPK as well as any typical N subject. In fact our experimental results to some extent bear this out with a somewhat unexpected finding to be presented later. With this experimental design there was no practical way to substantiate the real pre-experimental chess knowledge of any NK subjects beyond knowledge of legality. The experimental design had the flaw of allowing for the possibility of "informed guessing" on the Post-Test in that just prior to it when subjects performed the Database Comparison Section they would discover that 9 White-to-move positions were won, while 8 Black-to-move

positions were drawn.

A more difficult problem in design was deciding just how to present subjects with the positions which were to be looked up in the database in order to determine executability. Hence it seemed most reasonable to request that subjects only perform the complete decoding process, from presentation of a position in algebraic notation to its minimax-optimal game-theoretic value, in the Worked Examples Section (II) of the experiment. By giving tutorial coaching where necessary and recording the number and time required by each subject for each worked example, it was possible to ascertain whether this very tedious process of encoding, looking-up, and "unmasking" was in fact feasible; in other words was the Clarke Advice Text executable? Therefore in Section III subjects were provided with Position Numbers in order to facilitate the task of looking up and unmasking. The possibility of testing subjects on-line was considered, but then the issue of inexperience, unfamiliarity, and alienation from VDU's and their keyboards could arise. Tables 6.1, 6.2, 6.3, and 6.4 are the Pre-Test for N subjects, the Clarke Advice Text, the Database Comparison Section, and the Post-Mortem Test.

6.2.1 Example of Position Retrieval from the Clarke Database

Position: WK:d1, WP:c3, BK:f8

Note: This position is Figure 2 in Clarke (1977) which he describes as, "... more interesting and causes all but the best players some trouble".

Taking the formula given in the text, and considering the possible ranks of these three pieces, we get:

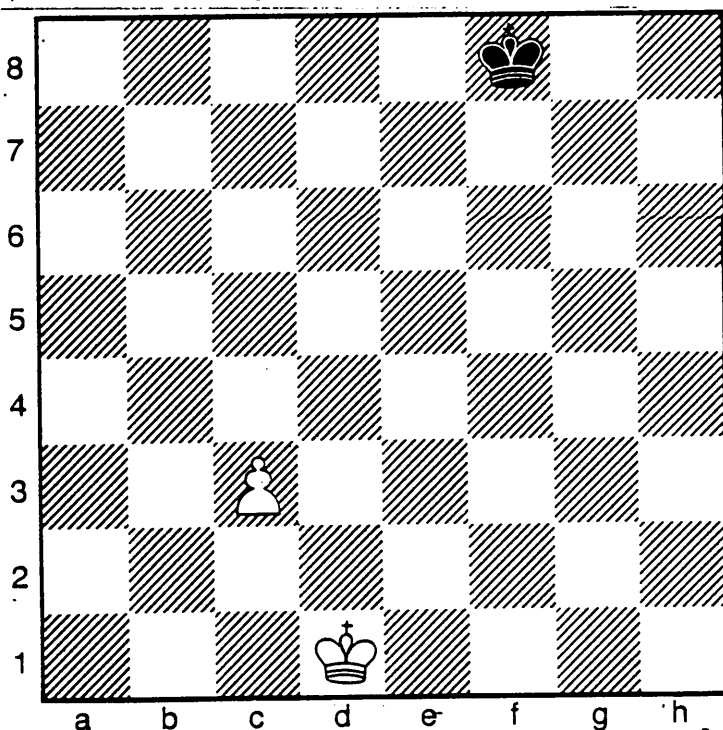
$$I = 16384 \begin{matrix} \text{WPR} \\ (3-2) \end{matrix} + 4096 \begin{matrix} \text{WPF} \\ (3-1) \end{matrix} + 512 \begin{matrix} \text{WKR} \\ (1-1) \end{matrix} \\ + 64 \begin{matrix} \text{WKF} \\ (4-1) \end{matrix} + 8 \begin{matrix} \text{BKR} \\ (8-1) \end{matrix} + (6-1)$$

giving  $I = 244829$

Since there are 18 entries per line we would find the entry representing the minimax-optimal value for this position by skipping  $24829/18 = 1379 \text{ rem.} 7$  or 1379 lines and 7 entries to the right. The entry for Position Number 24829 appears as 567.

$$567 = 27 * \begin{matrix} \text{W(I)} \\ 21 \end{matrix} + \begin{matrix} \text{B(I)} \\ 0 \end{matrix}$$

Which means White-to-move wins in (21-7) or 14 moves and Black-to-move draws.



### 6.3 Results

A total of 23 subjects, 12 N and 11 NK, were tested from 5th and 6th year classes at Holyrood High School in Edinburgh. All subjects ranged between 15 and 17 years in age with 5 of the 11 NK's and 1 of the 12 N's being females. The experiment normally lasted between 1 1/2 and 2 hours, with typically 1 to 3 subjects being tested at the same time. The approximate times required to complete the four parts of the experiment were as follows:

	Section: I	II	III	IV
<u>Subjects</u>	<u>(Time in Minutes)</u>			
N	Pre-Test 15	Worked Ex. 30	Clarke Adv. 30	Post-Test 15
NK	KPK Manual 30	" " 30	" " 30	" " 15

The average score on the Pre-Test (which consisted of 10 critical wins) of the N subjects was 5. Since this test involved the correct choice of one from two possible answers, Win or Draw, on chance guessing alone one could on average expect precisely this score, 5, with no a priori knowledge of the task domain. Thus we can assert that our novice subjects (N) were truly chess novices, at least with regard to KPK.

Of the entries in column 4 of Table 6.5 about half corresponded to unassisted performances, from which it was concluded that the Clarke database when used as an advice text for novice human subjects ranging between 15 and 17 years of age, was

sufficiently human-executable for our purpose. In worked examples subjects were able to carry out the necessary steps in order to obtain a Position Number from a position presented in algebraic notation, look-up the Position Value for this Position Number and unmask the position's minimax-optimal game-theoretic value from this number.

The ability of NK subjects to come to grips with the contents of the 4-page KPK Manual very much depended on their willingness to ask questions when unsure, their ability to retain and recall what they had read, and comprehension of the essential goals for each side in KPK. The Worked Examples Section (II) which all subjects participated in, normally required two fully completed efforts through the tedious process of converting a position in algebraic notation into a Position Number, looking-up the Position Value, and unmasking the Position Value into a minimax-optimal game-theoretic result with White-to-move and Black-to-move. Typically misunderstandings at this stage would occur because subjects did not realize that the symbols PR, PF, WR, etc. were to be substituted for by numbers and applied in the formula given for obtaining a Position Number. It was also necessary for the experimenter to explain the distinction between the Position Numbers in the first column of the database listing and the Position Values in the remaining 18 columns and that the first column was essentially a place-holder (finder) whose Position Value was located immediately following in the second column. It was noted that the time required to complete the



first worked example ranged between 8 and 20 minutes and was often twice as long as the time required for the second example. In some instances it was clear that subjects had a solid grasp of the task after only one worked example.

In Section III (Clarke Advice Text, Table 6.3) involving the exercise of looking-up the Position Numbers provided on a work sheet after subjects had put down their opinions (Win or Draw), the database's answers coincided with subjects' opinions on 73% of the instances recorded in column 7 (under 'O') of Table 6.5. This was somewhat unexpected since subjects had not as yet been given any coaching on the specific concepts involved in KPK: there was, however, the possibility of some indirect assimilation occurring during the training with the worked examples of the look-up procedure. In this regard subjects were asked to consider after each look-up how their answers differed or compared with the database's answers and why. Subjects' average Value score (of 10) was 7.38 as given in Column 8 under "V" of Table 6.5. This indicates that most of the time they were able to execute the look-up task, but were far from accurate in doing so. Roughly the same conclusions can be drawn from the Results (Column 9, under "R") of subjects' success in decoding minimax-optimal game-theoretic values with White-to-move and Black-to-move from the database. In short they were just as good at guessing the correct game-theoretic value "blind" as they were at looking it up in the Table. Conclusion: as a "crib" the Clarke Advice Text was ineffective, although the comparison of the last two columns of Table 6.5 (see also later) indicates a tutorial effect of its use.

The random probability of successfully guessing for the above part of the experiment was small. It can be calculated based on the number of wins and draws (11 and 9 respectively) in the 10 configurations (20 positions) to be looked up (with White to move and Black to move) and the corresponding expected proportion of correct answers given by a random retriever. This proportion is  $10.38 / 20$  or 51.9% while the actual results give 73% under Column 7 ('O') and 75.45% under Column 9 ('R'). Furthermore, matching answers under Column 'R' were only given credit if they included the correct minimax-optimal as well as game-theoretic values, e.g. "White wins in two moves". The likelihood of such precise, completely correct responses sheerly by guessing is therefore very remote.

The first real opportunity to determine whether any form of learning had taken place arose with the testing of all subjects in the Post-Test. Table 6.5 indicates the results on the Pre-Test and Post-Test for the appropriate subjects. One unexpected result was the extreme similarity of average scores on the Post-Test between the N (Avg. 6.42) and NK (Avg. 6.64) subjects. In only one case did a subject score less than 5 on the Post-Test and in this case his score still went up from 3 to 4. As Table 6.5 indicates, in two cases N subjects' scores were unchanged (D. Allen 5,5 and E. Regan 7,7) and in two cases subjects' scores went down, though it is worth noting that these went down from rather high initial scores (M. Tweedie 7,6 and D. Torrance 9,7). All NK subjects scored at least 5 on the Post-Test.

"Student's t-test", was performed to determine the significance of the mean Pre-Test - Post-Test difference of the 12 N subjects, and gave  $P < 0.05$  ( $P < 0.02$  for a 1-tailed test). With the logit transformation the significance was similar (see Table 6.6).

However it is clear from questioning to and from the experimenter that in some cases NK subjects had forgotten or misunderstood some of the rules such as Kings cannot move adjacent to each other, or that the BK could not move into a "checking square" of the P, or that the P could be captured by the BK if he is adjacent to it.

#### 6.4 Conclusions and Discussion

The Clarke database when used as an advice text for novice human subjects ranging between 15 and 17 years of age, was sufficiently human-executable as indicated by their performance on worked examples. The results indicating an overall average of over 70% correct decisions, (O = 73%, V = 74%, R = 75%; see Table 6.5) at least comparable with other more chess-knowledgeable

Subjects who had no previous knowledge of the legal moves and objectives for KPK were able to learn these within about one-half hour. Most significantly it was clear that subjects had learned something about KPK in order to have been able to obtain the correct value (win or draw) for a White-to-move position in 65.3% of the instances tested. The difficult problem to resolve is just what kind of learning had taken place. The only information akin to feedback which subjects had received on their knowledge of KPK were the results of the worked examples and the database's answers compared to their own opinions, it seems reasonable to assume that only some form of Skinner-type rote learning had occurred, with some generalisation, if only of the "geometrical displacement" type. This means subjects may have been able to recall some outstanding or overall piece configuration and its value from earlier examples (particularly when relating Section IV, the Post-Test, to Section III, the examples using the Clarke Advice Text).

However we can be quite certain that no concepts such as "if the WK is able to gain the opposition ahead of the P, then White wins" or "if the BK can reach the double-blockade square, then Black draws" had been learned or induced, nor had some facsimile of them. In other words subjects had not acquired any "deep understanding" of the problem domain and herein lies the distinction between concept learning and other forms of learning for domains of sufficient complexity.

The fact that in some cases subjects had been able to get correct answers on the Post-Test, despite the occasional serious misunderstandings (as mentioned in the previous Section (6.3)) which they had endured throughout the course of the experiment, can only be explained by their ability to recall some geometrical configuration from rote learning.

Pre-Test of Novice (N) Subjects on 10 Critical KPK Positions

(All Positions White To Move)

	<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>RESULT</u> ( <u>Win</u> <u>or</u> <u>Draw</u> )
1.	d3	d6	b3	
2.	c4	c6	c2	
3.	f4	a6	c5	
4.	e5	e7	c5	
5.	b2	d5	b3	
6.	a4	b7	b3	
7.	b4	a6	c4	
8.	g7	e8	c2	
9.	c7	a2	c3	
10.	d5	c3	a2	

Table 6.1

Symbols, Formulas, and Encodings for Clarke Advice Text

PR is the White pawn's rank, range 2-7  
 PF " " " " file, " 1-4  
 WR is the White king's rank, " 1-8  
 WF " " " " file, " 1-8  
 BR " " Black king's rank, " 1-8  
 BF " " " " file, " 1-8

Counting from 0 there are 98303 configurations in KPK. Each of these configurations is mapped into a Position Number (0 - 98303). The formula for obtaining a Position Number is:

$$\text{Position Number} = (16384 * (\text{PR} - 2)) + (4096 * (\text{PF} - 1)) + (512 * (\text{WR} - 1)) + (64 * (\text{WF} - 1)) + (8 * (\text{BR} - 1)) + (\text{BF} - 1)$$

Each of these Position Numbers has a one to three digit Position Value which is the encoded value of any KPK position with White and Black to move given by:

$$\text{Position Value} = (27 * \text{Value with White to Move}) + \text{Value with Black to move}$$

Thus by dividing the Position Value by 27 you obtain a further encoding which is the Value of the position with White to Move and the remainder is the Value of the position with Black to move.

The meaning of these encodings in chess terms is given as follows:

<u>Position With WTM</u>		<u>Position With BTM</u>	
Encoding	Meaning	Encoding	Meaning
0	Illegal, B in Check	0	Draw by repetition
1	Draw by repetition	1	Draw
2	Draw <-- Black Captures P -->	2	Draw in one move
3	" " in one move	3	Draw in two moves
4	" " in two moves	4	" " three "
5	" " in three "	5	" " four "
6	" " in four "	6	" " five "
7	" " in five "	7	" " six "
8	White Wins	8	Black Loses in 1 move
9	" " in one move	9	" " in 2 moves
10	" " two moves	10	" " in 3 moves
11	" " three "	11	" " 4 "
12	" " four "	12	" " 5 "
13	" " five "	13	" " 6 "
14	" " six "	14	" " 7 "
15	" " seven "	15	" " 8 "
16	" " eight "	16	" " 9 "
17	" " nine "	17	" " 10 "
18	" " ten "	18	" " 11 "
19	" " eleven "	19	" " 12 "
20	" " twelve "	20	" " 13 "
21	" " thirteen "	21	" " 14 "
22	" " fourteen "	22	" " 15 "
23	" " fifteen "	23	" " 16 "
24	" " sixteen "	24	" " 17 "
25	" " seventeen "	25	" " 18 "
26	" " eighteen "	26	" " 19 "

Table 6.2

Ten Clarke Advice Text Positions

Your Opinion ( 'W' or 'D' )	<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>POSITION NUMBER</u>	<u>Value</u>	<u>Result</u>
WIM BTM	1.a6		f5	d3 31789		WIM BTM
-----	2.a5		d6	a3 18475		-----
-----	3.b2		d6	b3 21099		-----
-----	4.d3		c6	b2 5354		-----
-----	5.c4		d6	b4 38571		-----
-----	6.a6		d5	c3 27171		-----
-----	7.e8		a7	c4 44848		-----
-----	8.b5		b7	d5 63601		-----
-----	9.c3		c7	d3 29874		-----
-----	10.h4		h6	d2 10223		-----

Table 6.3

WIM = White to move positions  
BTM = Black to move "



Post-Mortem Test of KPK

(All Positions White to Move)

	<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>Result</u>
1.	g6	b5	d3	
2.	a6	d7	a4	
3.	c2	e7	c3	
4.	e3	d6	c2	
5.	c4	b6	d4	
6.	b6	e5	d3	
7.	e7	a6	c3	
8.	f5	f7	d5	
9.	e3	e7	d3	
10.	f4	f6	b2	

Table 6.4

	NAME	AGE	TYPE (N or NK)	WORKED No.	EXAMPLES Time	CLARKE V	ADV. R	TEXT. R	PRE- TEST	POST- TEST
1.	D. Allen	15	N	2	8,17	N/A	9	14/14	5	5
2.	M. Gilmartin	16	N	2	17,10	N/A	0	3/18	6	8
3.	J. Binks	16	N	2	15,7	N/A	2	20/20	3	4
4.	P. Higgins	16	N	2	13,8	N/A	10	19/20	3	5
5.	B. McLeish	16	NK	1	20	N/A	2	3/20	(NK)	6
6.	M. Ewing	16	N	2	11,6	.90	9	20/20	1	5
7.	S. Turner	16	NK	3	8,11,10	.82 <sup>‡</sup>	0 <sup>+</sup>	10/20	(NK)	9
8.	K. Mills	17	NK	2	8,8	.70 <sup>‡</sup>	9	18/19	(NK)	5
9.	J. Mathews	16	N	2	10,8	.80 <sup>‡</sup>	4	20/20	3	8
10.	A. Meiklejohn	17(F)	NK	2	15,10	.75	10	20/20	(NK)	6
11.	S. O'Donnell	17(F)	NK	2	18,10	.70	9	16/20	(NK)	7
12.	E. Regan	16	N	2	17,16	.60	9	18/20	7	7
13.	C. McCarran	16(F)	N	2	13,6	.50	10 <sup>‡</sup>	20/20	3	5
14.	M. Ryan	17	N	1	10	.85	0 <sup>‡</sup>	18/20	6	9
15.	M. Tweedie	16	N	2	20,10	.80 <sup>‡</sup>	8	19/19	7	6
16.	D. Torrance	16	N	1	13	.90 <sup>‡</sup>	9	20/20	9	7
17.	T. Moan	15	N	1	13	.60	9	19/20	7	8
18.	L. Hewitt	15	NK	2	23,13	1.00 <sup>‡</sup>	9	18/19	(NK)	7
19.	N. Martin	16(F)	NK	1	8	.9	0 <sup>+</sup>	0/0	(NK)	5
20.	S. Jameson	15	NK	1	8	.45	0 <sup>+</sup>	8/18	(NK)	6
21.	W. Gilhoolley	15(F)	NK	2	20,8	.78 <sup>‡</sup>	0 <sup>+</sup>	15/20	(NK)	6
22.	J. Welsh	15	NK	2	20,8	.75	0 <sup>+</sup>	14/20	(NK)	9
23.	K. Moohan	15(F)	NK	1	30	.44 <sup>‡</sup>	0 <sup>+</sup>	5/20	(NK)	7
Tot.: 12 N, 11 NK					Avg.: .73	7.38	.75	5	5	6.53
Post-Test Avg.: N = 6.42, NK = 6.64										

Key: (E) - female subject

- O - proportion of correct matches (of 20) of subjects' opinions to results.
- V - number of correct uses out of 10 of numerical look-up code.
- R - retrieval-accuracy; the proportion of attempted retrievals from database which gave correct answer.
- \* - based on 9 responses; † - no responses given; ‡ - based on 17 responses;
- & - based on only 10 responses; p - incomplete responses; # - " 18 "
- L - based on 10 or more responses.

- Notes:
1. Worked Examples (Columns 5 and 6) here refer to correct completion of subject's task with or without experimenter assistance.
  2. Pre-Test and Post-Test (Columns 10 and 11) refer to number of correct responses from 10.
  3. NK subjects were not administered the Pre-Test.

Table 6.5 Results of 23 Subjects on Test with Clarke Advice Text.

Table 6.6 "Student's t-test" on performance of N subjects on Pre-test and Post-Test.

Subjects (n)	x (difference between Pre-Test and Post Test)	Sum Squares of Mean Difference
1	0	2.02
2	+2	0.34
3	+1	0.18
4	+2	0.34
5	+4	6.66
6	+5	12.82
7	0	2.02
8	+2	0.34
9	+3	2.50
10	-1	0.18
11	-2	12.82
12	+1	0.18

Mean ( $\bar{x}$ ) = 17/12 = 1.417  $\approx$  1.42       $\sum (x-\bar{x})^2 = 40.40$

$t = \bar{x} * \text{sqrt}(n) / s$  where  $\bar{x} = 1.417$   
 $n = 12$   
 The Standard Deviation,  $s = \text{sqrt}(40.40/(n-1)) = 1.92$

Hence  $t = 1.417 * 3.46 / 1.92 = \underline{2.55}$ .

Performing the logit transformation on the Pre-Test and Post-Test scores where  $\log\left(\frac{p_i}{q_i}\right) - \log\left(\frac{p_j}{q_j}\right)$  is used, gives x-values of: 0.00,+0.42,+0.19,+0.37,+0.95,+0.97  
 0.00,+0.37,+0.77,-0.19,-0.58,+0.23 .

which gives  $t = 0.29 * 3.46 / .46 = \underline{2.18}$  .

## VII THE NIBLETT ADVICE TEXT

### 7.1 Objectives

Using the advice strategy of Niblett (1982) which had been implemented as a PROLOG program and a technique of "structured induction", Shapiro and Niblett (1981) were able to derive decision rules which correctly classified all legal Black-to-move positions. From the above two works Niblett was able to produce a slightly more compact version of his earlier advice text reducing it to only five high-level rules, which determine whether a W-T-M position is won or drawn. It had several favourable features for purposes of testing human novice chessplayer subjects; that is:

1. The five rules were relatively very concise as compared to, for example, some of those in the Beal, Harris-Kopec, and Bramer advice texts.
2. The rules were virtually self-contained, not involving many other lower level attributes, and could be clearly defined by accompanying diagrams.
3. By a refined order of application Niblett was able to remove the necessity to test for special cases.

Experimentation sought to test the Niblett Advice Text with regard to:

- 1) executability
- 2) comprehensibility
- 3) memorizability

That is, in addition to performance with an external store of advice, as with the earlier open-book "cribs", performance was also tested employing an internal store of advice, i.e. the "closed-book", memorized advice text.

## 7.2 Design and Procedure of the "Open Book" Experiment

This experiment was designed to closely model the steps in the testing of the Clarke Advice Text (Chapter VI) wherever possible. Again there were to be two groups of subjects, with the NK subjects acting as controls. The 10 subjects in each group would be tested according to the following experimental design:

### I. Procedure for NK (Not Knowing chess moves) Control Subjects:

1. KPK Manual; experimenter assistance offered.
2. 10 stimulus Pre-Test (White-to-move (W-T-M), won or drawn) positions.
3. Grade Pre-Test; discuss wrong answers, allow subjects time to consider the results.
4. Post-Test on 10 W-T-M positions.

### II. Procedure for N (Novice) Subjects:

1. Pre-Test on 10 W-T-M positions (same as above).
2. Up to 5 Worked Examples; time, experimenter assistance required, and any remarks were all recorded.

3. Niblett Advice Text: 10 W-T-M positions.
4. Post-test on 10 W-T-M positions.

Thus NK subjects were again (as in Chapter VI) presented with the KPK Manual first to teach them the rules and objectives for KPK concisely in a few pages. They were permitted up to about thirty minutes to do this, and could ask the experimenter questions. All subjects were provided with a chess board marked from 'a' to 'h' below the bottom row of squares and from '1' to '8' up the left edge to facilitate use of algebraic chess notation. A White King (WK), a White Pawn (WP) and Black King (BK) were also provided for subjects' perusal throughout the experiment.

The first data obtained for all subjects (N and NK) was their performance on the Pre-Test which consisted of 10 positions, 7 won and 3 drawn, over about 20 minutes. All positions tested throughout the experiment were evaluated in terms of W-T-M. The Pre-Test was then scored for NK subjects with wrong answers gone over and discussed by the experimenter. The 10 position Post-Test (6 won, 4 drawn) was then administered to NK subjects with about 20 minutes again allowed. Their purpose as controls in the experiment was to obtain empirical evidence as a means of certifying that the domain, KPK, was sufficiently hard not to be solvable by some form of chance or rote learning alone. The N subjects were then presented with the Niblett Advice Text with 10-15 minutes allowed to read and study it followed by the

Worked Examples. Subjects could try any number up to all five of these Worked Examples for which the time required and experimenter assistance, if any, was recorded. The examples were then reviewed and subjects proceeded to the third phase of the experiment where the Niblett Advice Text was to be applied to 10 positions with up to 30 minutes allotted for the task. Of these 10 positions, 7 were won positions 4 of which required the application of the 5th rule (MAINPATT), 2 requiring the 4th rule (MAINPATT or RANK6), and 1 requiring the 1st rule (CANRUN), the remaining 3 test positions being drawn. The fourth and final phase of the experiment entailed the Post-Test (20 minutes) with most of the 10 positions related in terms of concept and configuration to those in the Pre-Test (Part 1) and Niblett Advice Text (Part 3) of the experiment presented earlier. The Pre-Test, Worked Examples, Niblett Advice Text, 10 test positions on it, and Post-Test, are presented in Tables 7.1 through 7.5 respectively.

The reader may note the change in design from Chapter VI with regard to proportioning of won and drawn positions in the test set. It was decided that the inclusion of a few (but less than 5) drawn positions in both the Pre-Test and Post-Test would serve better to prevent pure chance guessing, where previously this may have occurred and had been impossible to detect.

Pre-Test for N and NK Subjects on 10 KPK Positions (Niblett Advice Text)

(All Positions White To Move)

	<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>RESULT</u> ( <u>Win</u> <u>or</u> <u>Draw</u> )
1.	d3	d6	b3	
2.	c4	c6	c2	
3.	c1	e4	c3	
4.	e5	e7	c5	
5.	b2	d5	b3	
6.	a3	b7	b3	
7.	b4	a6	c4	
8.	g7	e8	c3	
9.	c7	a2	c2	
10.	d5	c3	a2	

Name: \_\_\_\_\_

Date: \_\_\_\_\_

N or NK: \_\_\_\_\_

Table 7.1



Worked Examples

For each of the following positions please indicate which of the 5 rules in the Niblett Advice Text applies to explain why White Wins (W) or otherwise indicate that the position is a draw (D). Assistance from the experimenter can be requested if required.

<u>Position (All White to move)</u>	<u>WIN</u>	<u>Rule Applicable (No.)</u>	<u>DRAW</u>
1) WK:b3 , BK:f7 , WP:d3			
<hr/>			
2) WK:f5 , BK:f7 , WP:c5			
<hr/>			
3) WK:a4 , BK:a6 , WP:g2			
<hr/>			
4) WK:g7 , BK:f3 , WP:a3			
<hr/>			
5) WK:g6 , BK:c7 , WP:e7			
<hr/>			

Name: \_\_\_\_\_

Date: \_\_\_\_\_

Table 7.2

The Niblett Advice Text

For any White-to-move position we can decide if he WINS by applying the following set of rules in order:

1. If CANRUN exists then White WINS.  
or
2. If the pawn's rank is 7 and the White King can safely move next to the Queening Square, then White WINS.  
or
3. If the pawn is a ROOKPAWN and if and only if a position is achievable where CANRUN exists or where ROOKPATT exists, White WINS.  
or
4. If the pawn's rank is greater than or equal to 5 and if White can achieve MAINPATT or RANK6 (or they exist) then White WINS.  
or
5. If MAINPATT exists or is reachable then White wins.

Otherwise

Black DRAWS

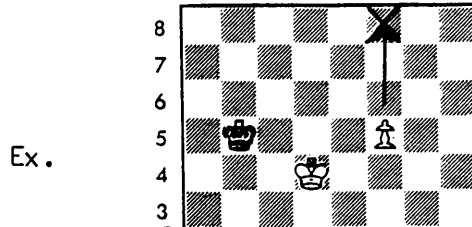
Table 7.3

(Page 1)

Definitions and Examples

1. CANRUN

exists if the positions is such that the pawn can safely advance to the Queening Square without moving the White King.

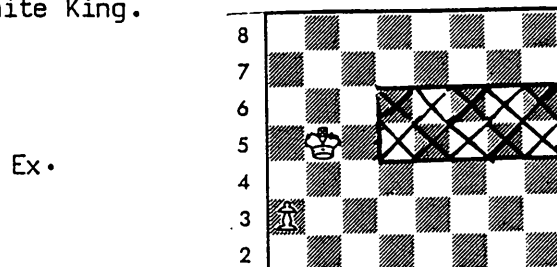


2. Queening Square

is the square on the 8th rank of the pawn's file, marked in the example above with an X.

3. ROOKPATT

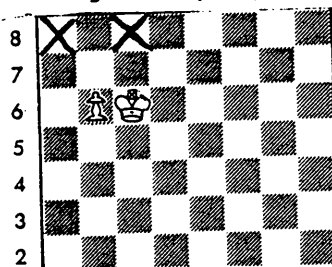
exists if the pawn is on the rook's (or 'a') file and the Black King is more files from the pawn than the White King and not more than one rank above the White King and not below the White King.



White King can 'box' Black King (on any square, 'X'), out.

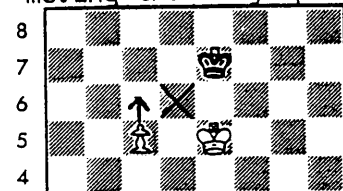
4. RANK6

The figure below with the Black King on any square but 'X'.



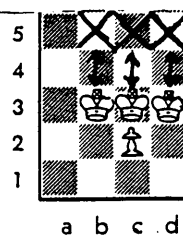
This pattern can be moved across horizontally.

Ex. RANK6 can be achieved by advancing the pawn, then moving the king up.



5. MAINPATT

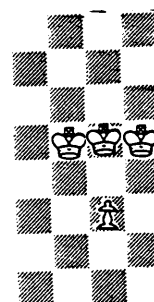
White to move in the figure below and the Black King is not directly opposite the White King (W) on a square marked 'X'.



These can be moved in any direction.

OR

White to move has the pattern below.



Niblett Advice Text 10 Positions

(All Positions White to Move)			<u>RESULTS</u>		
<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>WIN, RULE</u>	<u>or</u>	<u>DRAW</u>
1. d1	f8	c3			
2. c6	a8	b5			
3. d6	c4	a3			
4. d3	d7	c3			
5. h3	h7	b3			
6. f4	c6	d3			
7. c4	c6	d2			
8. c5	b8	d6			
9. a3	c6	b3			
10. e4	e6	d3			

Name: \_\_\_\_\_

Date: \_\_\_\_\_

Table 7.4.

Post-Mortem Test of KPK (Niblett Advice Text)

(All Positions White to Move)

	<u>WK</u>	<u>BK</u>	<u>WP</u>	<u>Result</u> ( <u>Win</u> <u>or</u> <u>Draw</u> )
1.	d1	b4	d3	
2.	a6	d7	a4	
3.	c2	e7	c3	
4.	e3	d6	c2	
5.	c4	b6	d4	
6.	b4	d6	d3	
7.	c5	c8	c6	
8.	f5	f7	d5	
9.	e3	e7	d3	
10.	f4	f6	b2	

Name: \_\_\_\_\_

Date: \_\_\_\_\_

N or NK: \_\_\_\_\_

Table 7.5

### 7.3 Results of the Open-Book Experiment

The 20 subjects who participated in this experiment were again members of 5th and 6th year classes at Holyrood High School in Edinburgh. All subjects ranged between 16 and 18 years in age, with 7 of the 10 NK's and 2 of the 10 N's being females. For the NK subjects the experiment ordinarily required about one hour in total for study of the KPK Manual, completion of the Pre-Test, its grading, and finally the Post-Test. The N subjects typically required about two hours to complete their four tasks with the time allotments (in minutes) approximately as follows:

1. Pre-Test (20)
2. Study Advice Text and Worked Examples (10-25)
3. Niblett Advice Text (35).
4. Post-Test (20).

Normally 2 to 4 subjects participated in the experiment simultaneously.

Tables 7.61 and 7.62 give the results of the experiment for both groups of subjects. All 10 N subjects completed the 5 worked examples in total times ranging between 10 and 25 minutes, with 3 subjects requiring assistance from the experimenter on example 4. A cursory analysis of the data indicates the following clear conclusions on the Niblett Advice Text when presented in open-book form:

- 1) The advice text was sufficiently human-executable. This is certified by the fact that 9 of the 10 subjects got 5 or more right answers for the right reason (RR) with an

Performance of N Subjects

Subject	M/F	Age	Pre-Test	Niblett Advice		Txt X	Post-Test	Net
				RR	RW			
1. D. Christie	M	18	7	8	0	2	9	+2
2. P. Angelosanto	M	17	9	7	0	3	4	-5
3. J. McKeen	M	16	9	7	0	3	6	-3
4. A. Meadows	M	16	3	5	1	4	4	+1
5. J. Hay	M	18	5	5	2	3	6	+1
6. F. Mitchett	F	17	5	4	1	5	4	-1
7. I. Pullin	M	17	5	6	2	2	7	+2
8. V. Creamer	F	16	7	7	1	2	7	=0
9. M. Devine	M	17	8	5	1	4	6	-2
10. J. Lawson	M	18	4	8	0	2	6	+2
<hr/>								
Tots. & Avgs.:	8M,2F	17	6.2	6.2	0.8	3.0	5.9	-0.3

Key: RR = Right Value, Right Rule  
 RW = Right Value, Wrong Rule  
 X = Wrong Value  
 Net = Net change in score from Pre-Test to Post-Test

Table 7.61 Results of N Subjects on Niblett Open-Book Advice Text

Performance of NK Subjects

Subject	M/F	Age	Pre-Test	Post-Test	Net Change
1. C. Anderson	F	17	8	8	=0
2. M. Figuerola	M	17	6	5	-1
3. G. Figuerola	M	17	9	8	-1
4. T. Hutchinson	F	16	3	6	+3
5. P. Murphy	F	17	7	5	-2
6. D. Burn	F	17	5	9	+4
7. P. McConville	M	16	5	4	-1
8. A. Sharkey	F	16	6	5	-1
9. A. Langton	F	16	4	4	=0
10. J. McLaughlin	F	16	4	5	+1
<hr/>					
Tots. and Avgs.:	3M,7F	16.5	5.7	5.9	+0.2

Table 7.62 Results of NK Subjects on Niblett Open-Book Advice Text

overall average of 6.2. Furthermore, 6 subjects were able to obtain 1 or 2 additional right answers each for position values (White wins or draw) through partially correct applications of the advice text.

2) The advice text did not in general or on average result in an improvement in score from Pre-Test to Post-Test. However, in each of the 5 cases where there was an improvement in score from Pre-Test to Post-Test there was also a score of greater than or equal to five in correct (RR) application of the Niblett Advice Text, with a total "Rightness score" (composed of RR and RW) ranging between 6 and 8.

3) As indicated by the results in Table 7.62 NK subjects did not improve from one test to the next (no intervening Advice Text use).

#### 7.4 Design and Procedure of the "Closed Book" Experiment

This experiment was quite similar in construction to the testing of N (knowing-moves-novices) subjects in the Open-book Experiment (7.2), except for the addition of a 15 to 30 minute time slot for memorization of the Niblett Advice Text and then its application to the 10 test positions but now in closed-book form. Therefore the complete experiment lasted about 2 1/4 hours. Since there were no control subjects (NK) used in this



experiment, extra caution was taken to ascertain that each subject was in fact a chess novice who knew the legal moves and rules required for this ending. The objectives for White (i.e to win by safely reaching the 8th rank with his pawn) and for Black (i.e to draw either by capturing the Pawn with his King, reaching a stalemate position, or by three-fold repetition) were reviewed in general before administering the Pre-Test. Thus the experimental procedure with time allotments in minutes (bracketed) was as follows:

1. Review legal moves, rules and objectives for KPK (15)
2. Pre-Test (20)
3. Study of Niblett Advice Text (15)
4. Worked Examples (12-26)
5. Memorization of Niblett Advice Text (15)
6. 10 Test Positions on Advice Text (20)
7. Post-Test (20)

The test materials used in all parts of the experiment were therefore identical to those used in the open-book experiment described in Sections 7.2 and 7.3 and illustrated in Tables 7.1 through 7.5. When performing the test on the 10 positions with the closed-book Niblett Advice Text some subjects jotted down a few notes of what they had memorized.

#### 7.5 Results of the Closed-Book Experiment

A total of 15 subjects, (8 males, 7 females), all students

at James Gillespie's High School in Edinburgh, were tested over a four week period. All subjects were 16 or 17 years of age. Subjects were recruited by Mr. McDougall, the Assistant Head Teacher, and normally 4 at a time participated in the experiment, with the exception of one session when a female N subject had to be screened out when it became apparent that she did not have an adequate grasp of the legal moves and rules of chess.

As indicated by Table 7.72, improvement from Pre-Test to Post-Test was observed to a degree that is formally significant to a level somewhat above earlier tests. ( $P < .02$  on a one-tailed test). Although expected scores from pure guessing alone on Pre-Test and Post-Test (5.8 and 5.2 respectively) do not greatly differ the actual average results, as in the Open-book experiment, the sum of the rightness scores (RR + RW) on the Niblett Advice Text, (7.26), is well above the expected "random" score of 5.8 .

Name	M/F	Age	Pre-Test	Niblett Advice		Txt	Post-Test	
				RR	RW	X	Test	Net
1. V. Notley	F	16	5	5	3	2	6	+1
2. G. Mercer	F	16	7	6	2	2	5	-2
3. R. Linsay	M	16	5	8	0	2	10	+5
4. A. Sheffield	M	17	5	8	0	2	6	+1
5. G. Swan	M	16	6	4	1	5	7	+1
6. J. Gardiner	M	16	4	5	2	3	3	-1
7. R. Patton	F	16	4	7	1	2	5	+1
8. C. Ritchie	F	16	4	5	2	3	7	+3
9. N. Paterson	M	17	5	6	0	4	7	+2
10. D. Neilson	M	17	3	4	3	3	9	+6
11. D. Carnall	M	16	5	3	3	4	5	0
12. C. Murray	F	16	4	7	0	3	5	+1
13. M. Carson	F	16	6	6	1	3	5	-1
14. D. Campbell	F	16	7	9	0	1	6	-1
15. R. Stewart	M	16	6	6	2	2	7	+1
Tots. & Avgs.: 8M,7F		16.3	5.1	5.93	1.33	2.73	6.2	+1.15

Key: RR = Right Value, Right Rule  
 RW = Right Value, Wrong Rule  
 X = Wrong Value  
 Net = Net Change in score from Pre-Test to Post-Test

Table 7.71 Results of N Subjects on Niblett Closed-Book Advice Text

Subjects (n)	x (difference between Pre-Test and Post Test)	Sum Squares of Mean Difference
1	+1	0.02
2	-2	9.80
3	+5	14.98
4	+1	0.02
5	+1	0.02
6	-1	4.54
7	+1	0.02
8	+3	3.50
9	+2	0.76
10	+6	23.72
11	0	1.28
12	+1	0.02
13	-1	4.54
14	-1	4.54
15	+1	0.02

$$\text{Mean } (\bar{x}) = 17/15 = 1.13$$

$$\sum (x - \bar{x})^2 = 67.78$$

$$t = \bar{x} * \text{sqrt}(n) / s \quad \text{where } \bar{x} = 1.13$$

$$n = 15$$

$$\text{The Standard Deviation, } s = \text{sqrt}(67.78/(n-1)) = 2.20$$

$$\text{Hence } t = 1.13 * 3.87 / 2.20 = 1.99$$

Table 7.72 "Student's t-test" for performance of subjects on Pre-Test and Post-Test of Experiment Using Niblett Closed-Book Advice Text.

## 7.6 Conclusions and Discussion

From the foregoing results of experimentation with the Niblett Advice Text we can conclude that for chess novices it is:

- 1) sufficiently executable
- 2) for the most part comprehensible
- 3) not too complex to be memorizable

A better understanding of its feasibility as a learning tool for KPK would require further experimentation. For example, it is unclear how much subjects depended on the results of their employment of the advice text (AT), open or closed-book, when taking the Post-Test. Had they reverted primarily to their knowledge of KPK prior to the experiment (which in most cases was minimal) or not? How many subjects were simply confused by the AT? How often was the AT executed partially correctly (i.e. inferior moves were made in their analyses) but leading to a totally correct scoring (RR) of their answers?

Such difficulties depending on numerous variables and unknowns will always become involved when one attempts to measure learning, but this has not been the main issue here. Instead we have been interested in the question of the executability, comprehensibility, and memorizability of a given knowledge representation. It is clear that performance on the open-book experiment did not greatly differ from that on the closed-book

experiment. In both cases we may conclude by inspection of Rightness scores that the AT was executable. Furthermore, in both instances where there was a tendency for improvement from Pre-Test to AT performance, there was also an upward trend from AT to Post-Test. We may therefore conclude that when the AT was comprehensible and subjects correctly learned new concepts concerning the KPK domain it was reflected by an improved Post-Test score.

Performance using the closed-book AT as an internal store of advice was generally superior to performance with the open-book AT as an external store of advice. This is indicated across the line in Rightness score (compare Tables 7.61 and 7.71), by change in average scores from Pre-Test to AT, and change in average scores from Pre-Test to Post-Test.

## VIII OVERALL SUMMARY OF CONCLUSIONS

A specialized sub-domain of chess, KPK, has been studied from many points of view. In Chapter I of this thesis we consider how the chess master views this domain and describe efforts to program it successfully. Chapter II focuses on the Harris KPK Program, a standard algorithmic program, detailing efforts to verify and improve its correctness; Chapter III is a description of Beal's and Bramer's programs along with Clarke's Database. Chapter IV compares the four programs as implemented in the same language, Algol-60, on the same machine, DEC-10. Chapter V describes early experimental work and the translation process for the Beal, Harris and Bramer Programs, as well as providing the actual advice texts. The results, summary and conclusions from this experimentation are also given here. Chapter VI and VII are comprised of the account and results of further experimentation with the Clarke and Niblett Advice Texts respectively.

The work detailed in Chapter II gives evidence for the difficulties encountered in debugging a KPK program developed by conventional algorithmic methods. The proliferation of special cases and the necessity of checking exhaustively for maladies which may have resulted by side-effect from corrections to them is largely responsible for these difficulties.

Five correct or nearly correct knowledge representations for KPK have been studied primarily with regard to the following three properties:

1) computational efficiency, which in order from highest to lowest (Niblett excluded) resulted in: Beal, Harris, Bramer, Clarke.

2) cognitive efficiency, considered with regard to five correct or very nearly correct computer program representations for KPK translated into English language "advice texts" distinguished as:

1. Niblett            5 patterns, very little calculation
2. Harris-Kopec    7 patterns, much calculation
3. Bramer            19 patterns, considerably less calculation
4. Beal                48 patterns, little calculation
5. Clarke            98304 patterns, considerable calculation

Though cognitive efficiency was experimentally unresolved in tests with Beal, Bramer, and Harris-Kopec, it was apparent that none of these was very easy to use or ideal. Indications of intelligibility were that Clarke and Beal were too extensional, Harris-Kopec too intensional, while Bramer and especially Niblett were sufficiently human-intelligible.

3) grain size differences and the necessary tradeoffs between storage-space and execution-time to fall within the bounds of the human window.

Experimentation was also performed to determine the executability and memorizability (the latter only in the case of the Niblett Advice Text) of these advice texts.

## IX CASE STUDIES FROM REAL WORLD APPLICATIONS

The relationship of the experiments discussed in the foregoing sections to real world problems is hereby considered. The problems our human KPK subjects confronted in attempting to correctly carry out their tasks under idealised laboratory conditions may be viewed as a microcosm of those faced by operators of complex systems (such as air traffic control centres and nuclear power stations) in emergency situations. As an example consider when an operator attempts a corrective action (by analogy the subject searches for other legal moves or solutions from a given problem position) and receives no intelligible feedback, or possibly even conflicting information, as to whether the action has brought him closer to a correct solution to the problem. The same again was true for the human subjects here -- e.g. once embarking on a legal move from an initial won position with White to move, subjects had no way of knowing whether this path of analysis would lead them closer to a correct solution (i.e. a win), nowhere (i.e. cycling), or seriously astray (i.e. a move which changes the game-theoretic value from a win to a draw). A correlate of the relative opacity to them of all but the Niblett and the Bramer advice texts was that they could not obtain guidance by consulting the machine's own representations.

In sum, there are two special properties which are shared by the laboratory case studies testing KPK advice texts on human chess novices and by the operators of computer-controlled complex systems:



1. codified both within these control programs for complex systems and also within the advice texts are domain-specific heuristics which are beyond the comprehension of those who operate or use them.
2. the "99% effect" — while under normal conditions of performance these advice texts or complex systems may be transparent to their user or operator, under conditions of breakdown (the unusual "1% cases") the opacity of machine-oriented rules or corrective actions becomes a critical factor compounding these conditions.

We now address the issue of Problems of the "Human Window"

-- the need for satisfactory conceptual interfaces between man and machine in the broader context of information technology. This issue has pervasive technological consequences in software design, and will now be discussed in the following Sections in relation to four documented breakdowns of the user interface in complex computing systems:

- (9.1) Three Mile Island
- (9.2) Air Traffic Control
- (9.3) NORAD Military Computer
- (9.4) Royal Dutch Steel

Investigations have been carried out into the above four particular cases as examples of "user-inscrutability" of complex and sophisticated systems. In each case automation, the use of machinery to perform tasks that previously had been accomplished exclusively by humans, plays a key role. In the past two decades automation has been inherently related to computerisation as has been the case in the areas which have been studied here. Since at least for the foreseeable future man intends to be in charge of communications with machines, and not vice-versa, it is necessary that the relationship between man and machine be humanised rather than further "computerised".

The mismatch between a technological system and the humans who operate it can be either at a "surface" or "structural" lev-

el. A knowledge representation can be correctable at the surface level, but if its structure or conceptual basis is unsound, then no amount of human technology can correct it. The nature of the underlying causes of mismatch between man and machine, whether essentially at the surface level or the structural level, is the issue to which our investigation is oriented.

It is no longer just laymen who are unable to comprehend how computers and the advanced information technology under which they operate, can co-ordinate within large sophisticated systems; control rooms in nuclear power stations, in air traffic, and on oil platforms are instances where large systems are now generally beyond the technical and scientific sophistication of those who operate them.

### 9.1 Three Mile Island

The nuclear power station at Three Mile Island, Pennsylvania (known as T.M.I.-2 since it is one of two plants there) is one particular case in point. In its conclusions in "The Accident At Three Mile Island" the Report of the President's Commission summarizes the "Causes of the Accident" with (p11, bottom):

"In conclusion, while the major factor that turned this incident into a serious accident was inappropriate operator action, many factors contributed to the action of the operators, such as deficiencies in their training, lack of clarity in their operating procedures, failure of organizations to learn the proper lessons from previous incidents, and deficiencies in the design of the control room."

In this overview of the "Causes of the Accident" the Commission stresses that it was the "lack of attention to the human factor in nuclear safety" which was to blame for the seriousness of the accident. While the training of operators and operation of the control room may have been adequate under normal circumstances, they were seriously deficient under accident conditions. Operators, even senior ones, did not have a sufficiently deep understanding of nuclear power under the complex prevailing circumstances. The specific procedures which operators had to follow as a result of the accident were at the least very confusing, and could reasonably have been interpreted to lead to the actions which the operators had mistakenly taken. Furthermore, the lessons from previous accidents (which might have prevented the T.M.I.-2 accident altogether) did not result in new clear instructions being passed on to the operators.

The control room also proved lacking in many ways, particularly with regard to the "human interface". The commission report continues:

"The control panel is huge, with hundreds of alarms, and there are some key indicators placed in locations where the operators cannot see them. There is little evidence of the impact of modern information technology within the control room."

Let us now consider another area of growing concern for information technology.

The year 1980 may be labelled by historians and scientists as the "Year of Mishaps" for American technology. Instances in the domain of control and operation of civil and military aviation alone, are most evident. Some highlight examples are:

- (1) The numerous near-misses at Kennedy and other busy national and international airports.
- (2) The technological problems with the operation of the Hercules Sea Stallion helicopters which resulted in aborting the rescue mission to free the American hostages in Iran.
- (3) The NORAD military computer which falsely signalled a Soviet nuclear attack on three separate occasions.
- (4) Numerous crashes of fighter planes on training missions.

These incidents (cases (1) and (3) will be among those investigated more specifically) dramatise the idea that as technology rapidly advances and computers are given more responsibilities, the role of the man-machine interface becomes more critical.

## 9.2 Air Traffic Control (A.T.C.)

An article in The New Scientist (17 July, 1980) entitled "Near Misses in the Sky", focuses on the problems of computers in air traffic control. We cite some examples and summarize its main points.

On 31 October, 1979 at 6.52 P.M. a faulty connection in one memory element in the IBM-9020 computer at the air traffic control (ATC) centre in Leesburg, Virginia nearly resulted in a mid-air disaster. The computer went down for 6 minutes during which two planes, a northbound Boeing 737 and a southbound Delta Airlines Lockheed L-1011 some 320 km. away, at altitudes of 8700 and 9300 m. respectively, were heading for Wilmington, North Carolina. Then at 7:06, when the pilot of the L-1011 asked permission to descend to 8400 m. no-one at the control centre noticed the potential hazard. It was only because the 737 pilot saw the descending Delta and managed to swerve sharply that a mid-air collision was averted.

A routine fault analysis seconds after this incident indicated that a malfunctioning memory component had attempted to contact each controller's display console. However, the memory component was part of a newly reconfigured system and had been misprogrammed to contact one more display console than existed. This mistake and the subsequent non-response from the "missing" terminal, resulted in the shutdown of the entire IBM-9020 system.

When this and other such incidents were reported to the U.S. Federal Aviation Administration (F.A.A.) which oversees A.T.C. centres as well as airport control towers, the response was: "no accident has ever been directly or indirectly associated with a computer failure."

Similar computer malfunctions have occurred more than 850 times in 1979 at the 20 ATC centres across the United States. After the particular incident above, ATC controllers complained, and one Leesburg controller stated: "This type of accident will happen again. We can't keep defending this machinery. If the equipment is obsolete, admit it and take corrective action."

Perhaps this air traffic controller has missed the main cause of his problems. It is not the failures of machine hardware which are entirely to blame -- it is the methodology of fault diagnosis and how these faults are conveyed to the humans in charge that is responsible. In this sense the correct functioning of the hardware is the syntax component of the system. However, once a fault occurs somewhere along the control system, the issue of semantics, just "what is going on" in humanly comprehensible terms, becomes the key. When aircraft identity and altitude data (which normally appear on the controller's display screen along with the aircraft's radar images) suddenly disappear, then the air traffic controller is plunged into a vacuum, the cognitive equivalent in an automobile of the discovery that the windscreen has fogged and the foot brake is not functioning.

The New Scientist article goes on:

"A few weeks later, on 25 November, the A.T.C. centre at Fort Worth, Texas, experienced a power fluctuation which blew 32 fuses. The computer system failed, leaving display screens blank, and depriving controllers of even raw radar data for 4 minutes. No near-misses were reported, but the mishap caused another failure in the I.B.M. computer. This led the system to shut down again on 28

November.

During this second failure, controllers at Fort Worth were following the progress of 19 aircraft, including two American Airlines Boeing 727's. Just before the breakdown, the controllers gave permission to one 727 to descend to 5400 m., the exact altitude at which the other other 727 was flying only 130 km. away. When the displays flickered back to life 4 minutes later, an alert controller saw the conflict and advised the descending airliner to slow down. The two aircraft passed with only 180 m. between them."

In 1979 the Leesburg centre alone had 200 failures, while Fort Worth had 74. New York's A.T.C. centre at Kennedy Airport had the longest breakdown on 14 November, 1979 lasting 13 hours. Fortunately there were no close calls, but the costs in tension, discomfort and dollars can be surmised when we consider the fact that 1.2 million gallons of excess fuel were burned.

The events reported above and many more incidents which were not recorded specifically as "near-misses" led the Professional Air Traffic Controllers Organisation (PATCO) to conduct a survey of computer failures in the F.A.A. air traffic control centres. The survey led to allegations at a Congressional hearing (April 9, 1980) that although the F.A.A. has \$3000 million to spend on computers, it persists in operating a computer system that is unreliable and "possibly senile". Just weeks before this hearing Congress received a report from the Government Accounting Office which concluded that the F.A.A.'s "past efforts to deal with mid-air collisions have been hampered by a lack of internal coordination and disagreements over policy, approach, timing and direction" (ibid., p189). These problems sound very similar to



those of the Nuclear Regulatory Commission and their role in directly or indirectly causing the accident at Three Mile Island.

Problems with computer breakdowns in A.T.C. were highlighted by two further incidents.

The first event occurred on 18 January, 1980. Gerald O'Brien, became the first controller ever to be charged by the F.A.A. of tampering with A.T.C. equipment. Allegedly he wilfully removed data from computerised radar scopes and thereby contributed to the potential endangerment of a Soviet Aeroflot jetliner approaching Kennedy International Airport. This particular Aeroflot flight included the Soviet Ambassador, Anatoly Dobrynin on it. The incident occurred at a time when the local chapter of PATCO was publicly opposing the handling of Soviet or Iranian planes, because of the Soviet intervention in Afghanistan and the holding of American hostages in Iran. After a shift change at Kennedy's Instrument Flight Rules (IFR) room, an alert supervising controller became aware that the Aeroflot jet's radar blip did not carry any computer-generated alphanumeric identifying data block, normally indicating airline flight number, altitude and ground speed. The supervisor then called the Air Route Traffic Control Centre (ARTCC) to clarify the situation. The fact that there was another plane in the same air sector as the Aeroflot led to a confused instruction from the supervisor telling the Aeroflot to descend 10 miles too early in the normally very busy air space over Long Island. Fortunately there was no other traffic in the area so that no serious danger was present.

The second incident was just one of a series of close calls during the summer months at New York City's airports. On 9 July, 1980, about 10 miles east of Kennedy, a British Airways Boeing 707 and a small private twin-engine Cessna passed within 150 m. of each other. This was not directly caused by any computer or human error, despite the fact that numerous mechanical breakdowns and human operational errors had been occurring at just this time. The computer in the IFR room did not automatically provide the British plane's data block, next to the plane's "blip" on a radar scope, but this was obtained manually.

The real problem lies in the crude "see and avoid" system intended to assure safe separation of controlled and uncontrolled aircraft mingling in the same airspace. Theoretically, under this system, since the weather was clear, the pilots of the planes involved should have been able to see each other and avoid danger. But the small craft was only detectable as a radar blip and was only tracked due to the initiative of a controller who gave specific instructions for the computer to obtain further data.

We can see that present methods of A.T.C. need to be carefully scrutinized, for in a number of cases disasters have been avoided only by astute human efforts beyond the call of duty. Not enough attention has been given to how operators must prevail in situations where computers malfunction. This is perhaps more confusing and hazardous than when humans can rely on radar and radio transmissions alone.

There is a clear split between long-term U.S. and European views of what the roles of people should be within A.T.C. In Hedley Voysey's article, "Problems of mingling men and machines" (New Scientist, 18 August 1977) the U.S. view is presented by a quote from Andres Zellweger of the Advanced Concepts Staff of the F.A.A.:

Today's A.T.C. system, which employs over 25,000 air traffic controllers, is overly labour intensive and, with the current traffic control procedures, will become even more so in the future. The F.A.A. plans call for increased automation of controller function with a human role change from controller of every aircraft to A.T.C. manager who handles exceptions while the computer takes care of routine A.T.C. commands.

The attitude and direction of U.S. A.T.C. is diametrically opposed to the European outlook. Peter Sturgeon, of the Applied Psychology Department of Aston University, has for some years been examining various aspects of the A.T.C. man-to-machine relationship. The European approach is aimed at a partnership (symbiosis) between man and machine, which he sees as superior to either working alone. Due to real doubts over the reliability of U.S. methods, European and U.K. controllers have been reluctant to adopt the U.S. approach. The main concerns of the Aston University group are:

- (1) Will the controller who has to intervene in an exceptional case be properly placed to do so?
- (2) Over long periods of time the sheer lack of verbal communication between A.T.C. and individual aircraft

may lead to mistakes in instructions or

- (3) a generally increasing reluctance of humans to intervene in the system at all.

The F.A.A. plans extend almost to the end of the century and thus it is unlikely that there will be a shift in the U.S. approach for some time.

### 2.3 NORAD Military Computer

Within an 8-month period during 1979-1980 the U.S. experienced 3 false alerts indicating that it had been attacked by Soviet missiles. These were all due to computer error. The first reported false alert occurred on 9 Nov., 1979 and was the result of a mechanical error when a "war game" information tape was inadvertently fed into "live channels" setting off early warnings of a nuclear missile attack from Soviet submarines probably located in the north Pacific. This meant that 10 jet interceptors from three bases in the U.S. and Canada were scrambled aloft and missile bases throughout the U.S. were put on low-level alert.

While the six-minute alert had been considered sceptically enough not to notify the President or Secretary of Defence, if it had lasted just one more minute the incident would have been brought to their immediate attention. There have been several such false alarms in the past, notably in the late 1950's and early 1960's caused by computer failures, natural phenomena, and test firings, but this was the first where the command went out

from the NORAD (North American Air Defence Command) centre in Colorado Springs, Colorado, to the vast complex of defence centres chained across the United States.

A second false alert occurred on 3 June, 1980, again within NORAD due to a computer error. Again the warning indicated a missile attack from the Russian mainland and from submarines. Within seconds the alert spread to the U.S. strategic air command in Nebraska and to the National Military Command Centre in the Pentagon.

It required only 90 seconds for the Command Staff buried 500 m. below Colorado's Cheyenne Mountain to check the alarm against radar and satellite information to confirm that there was actually no evidence of a Soviet attack. A minute later it was decided that the nuclear alert which had been transmitted to U.S. military command posts around the world, was cancelled. Thus the false alarm lasted only 3 minutes, but it required 20 more minutes for the U.S. strategic forces to stand down.

On 6 June there was yet another alert, but this was an intentional one, in an effort to duplicate the circumstances surrounding the first event on 3 June. As previously, the alert was not deemed serious enough to notify President Carter or Secretary of Defence Harold Brown, although the "situation room" in the White House and the President's command post were told.

A two-week investigation headed by Gerald P Dineen, an Assistant Secretary of Defence, revealed that the false alerts were

caused by a single faulty integrated circuit.

The key precaution which is built into the NORAD alert system is that it is an 8-stage process whereby a key human decision must be made at each stage. This prevents machinery alone from ordering a nuclear strike. Defence officials at Cheyenne Mountain gave the further assurance: "If there are eight different stages, then we were only at step one ..."

However, the real concern of U.S. officials are two circumstances which could be the direct result of a false alarm:

1) the "shrinking time factor" and 2) the possibility of "escalating responses".

1) the "shrinking time factor" is the critically short period of time in which humans in either the U.S. or Soviet Union must be able to read computer signals and make decisions. This is directly related to the fear that since the Soviet warning systems and technology are less sophisticated than those in the U.S., they would have less time to consume for making a decision and testing the certainty of their information. Perhaps even more pertinent to this shrinking time factor is that approximately 75% of the Soviet nuclear strike force is on land-based launchers. This means that they are most vulnerable targets for U.S. attack and that they require more time to reach their targets as compared with the more balanced land-based, airborne and seaborne force in the United States.

2) The possibility of "escalating responses". Though these nuclear alerts were discovered to be "false" and the result of some computer error in 6 and 3 minutes respectively, it is quite significant that 20 minutes were required to bring U.S. forces down from their higher state of alert. An alert, whether or not false, means that there is sudden, great activity at military command posts. Bomber crews start their engines, some planes even take off, land-based missile silos are brought closer to firing, and ballistic missile submarines receive signals. There is little doubt that the Russians could spot at least some of these movements; they could then respond quickly and with more magnitude. These "escalating responses" could continue until a full scale nuclear confrontation could result from a simple misunderstanding of normal precautionary steps due to a false alarm.

#### 9.4 Royal Dutch Steel

The study of automation and how it effects the operators of a large control system requires the cooperation of many different parties. At the highly automated Hoogovens hot strip mill of Royal Dutch Steel such a rare cooperation between management, psychologists, ergonomists and workers was achieved. The study was carried out by specialists of the British Steel Corporation, the Technical University of Delft and Hoogovens, together with production management for a period of 18 months from Feb., 1975.

The productivity of the Hoogovens hot strip mill had dropped abruptly and the key question was, to what extent was this problem caused by a newly installed highly automated control system.

The following summary of the main conclusion is from Hedley Voysey (New Scientist, 18 Aug. 1977, pp. 416-7):

"The operators became so unsure of themselves that, on some occasions, they actually left the pulpits used for control unmanned ... The operators also failed fully to understand the control theory of the programs used in the controlling computer, and this reinforced their attitude of "standing well back" from the operation - except when things were very clearly going awry. By intervening late, the operators let the productivity drop below that of plants using traditional control methods. So automation had led to lower productivity and operator alienation simultaneously."

An idea which had appealed to the plant's designers was to enclose the steel strips being rolled. However this seriously obstructed the ability of the plant's experienced operating staff to assess when there was a serious computer failure, either in the complex programs or electrical input sensors.

The remedy for this unsatisfactory situation at the Hoogovens plant was not simple either. It required that operators be taught the intricacies of the system they were running. This involves the control methods, the operation of programs in process terms, and all visual cues which might be helpful in detecting problems. With the complex physics of the system, the theory of the control system is intricate as well. Again, as in the case of T.M.I.-2, we see the need that operators are not just button



pushers who can only deal with a limited number of situations, but highly skilled and trained individuals, with appropriate salaries for their responsibilities.

We summarize some of the further conclusions of the Hoogovens study, which was entitled: Human Factors Evaluation Hoogovens No. 2 Hot Strip Mill.

- (a) There is no evidence that automation at the Hoogovens plant forced people into jobs which are socially unacceptable.
- (b) Automatic systems should be designed so that it is possible for the operator to anticipate potential problems and take preventive action, rather than react to problems only after they have already arisen.
- (c) Information displays should be designed to help the operator predict performance and to help him understand the decisions being taken by the automation, as opposed to the use of displays only to indicate the state of a process.
- (d) The visual and auditory information which the operators use in the present mill is of paramount importance to their task, and special attention should be given in future designs to providing the best possible view of the process.

- (e) A suitable form of back-up is required in the event the process computer is out of action or inaccurately set up.
- (f) The facilities for operator interaction with the computer system should be extended, particularly to enable him to monitor and improve the performance of the automation; this would include the alteration of incorrect or suspect data to produce a "better set up"; by "game playing" with the computer to examine the desirability of alternative courses of action; and by using, where necessary, the operators' corrections to the automatic control as well as automation feedback loops, to update the computer set ups.
- (g) Operator training should emphasize the function and performance of the automation. Operators should be trained so that they can define and articulate their technical needs clearly to production and automation engineers.

These conclusions are in line with the general European view of what the relationship between man and computer should be.

## 9.5 Overview

For certain special tasks which are socially critical, stand-alone information systems should not be entrusted with operational control. Certification should only be granted to systems which demonstrably conduce to the user's understanding of the task-environment. A clear distinction between "surface" (cosmetic) and "structural" (conceptual) causes of interface breakdown in an information system is therefore needed. The investigation of our four case studies from the real world leads to the common finding that some task-environments are still too infested with defects of interface design at the surface level for the study of the latter cause of breakdown to be even addressed. This theoretical case can be demonstrated in model domains on a laboratory scale such as we have investigated here, and must be kept in mind as real-life task-environments of increasing complexity are penetrated by machine systems.

The fact that of the five advice texts tested only the Ni-blett Advice Text was clearly "internalisable" (in the sense of becoming part of the user's mental furniture) gives some indication favouring the A.I.-structured approach for complex problem-domains.

An analogy is optical instrumentation where "surface" causes of poor viewing respond to cosmetic correction (e.g. polishing a clouded lens), whereas some other causes are structural (such as faulty focal adjustment) and require less superficial interven-

tion. In software today, the issue of structural as opposed to surface causes of inscrutability has yet to be addressed by the profession at large.

The case studies cited here, each involving computers in complex systems, but with failures at the interface between man and machine, provide only minimal evidence for the human window hypothesis for the reason given above, i.e. they incur issues which are too low-level due to opacity at the surface level. Nonetheless they point to hazardous trends and give suggestive evidence for what the required focus of future work should be.

Commitment must therefore be to interactive man-machine systems wherever possible. Within this context the only computer-based decision structures which can be regarded as safe are those amenable to conceptual debugging at run time. No commercially available software of the present state of the art has this property, with the exception of little-known packages of the "expert systems" type. These special A.I. systems can be extended to incorporate high-level expertise in fault diagnosis and correction both in other programs and also in themselves. We do not see any other path to ultimate safe solutions.

## REFERENCES

- Averbakh, Y. & (1974) Pawn Endings,  
Maizelis, I. London: Batsford.
- Beal, D. (1977) K + P vs. K chess endgames - discriminat-  
ing wins from draws. Unpublished Report,  
London: Queen Mary College. Also  
(1980) Appendix 5 of Beal & Clarke, The  
construction of economical and correct  
algorithms for king and pawn against  
king, in Clarke, M.R.B. (ed) Advances in  
Computer Chess 2, Edinburgh: Edinburgh  
University Press, 1-30.
- Bitner, J. & Hansche,  
B. (1976) Topics concerning the KPK endgame.  
Research Report, Urbana-Champaign:  
University of Illinois.
- Bramer, M.A. (1977) Representation of Knowledge for Chess  
Endgames: Towards a Self-Improving Sys-  
tem. Ph.D. Thesis, Milton Keynes: Open  
University.
- Bramer, M.A. (1977a) King and Pawn against King: using effec-  
tive distance. Technical Report, Milton  
Keynes: Open University, Faculty of  
Mathematics.
- Bramer, M.A. (1980) Correct and optimal strategies in game  
playing. The Computer Journal 23, No. 4,  
347-352.
- Bramer, M.A. (1980a) Representing pattern-knowledge for chess  
endgames: an optimal algorithm for King  
and Pawn against King. "Advances in Com-  
puter Chess 2" pp. 82-96. (Ed. M.R.B.  
Clarke), Edinburgh: Edinburgh University  
Press.
- Bramer, M.A. (1981) Machine-aided refinement of correct stra-  
tegies for the endgame in chess.  
Mathematics Faculty Technical Report,  
Milton Keynes: The Open University,  
Faculty of Mathematics.
- Bratko, I., Kopec, D. &  
Michie, D. (1978) Pattern-based representation of  
chess endgame knowledge. The Computer  
Journal, 21, No.2.
- Bratko, I. & (1980) An advice program for a complex chess  
Michie, D. programming task. The Computer Journal,  
23, No. 4, 353-59.
- Chase, W.G. &  
Simon, H.A. (1973) Perception in chess. Cognitive Psychology  
4, 55-81.

- Clarke, M.R.B. (1977) A quantitative study of King and Pawn against King. Advances in Computer Chess 1, pp. 108-118 (Ed. M.R.B. Clarke), Edinburgh: Edinburgh University Press.
- Clarke, M.R.B. (1980) The construction of economical and correct algorithms for KPK. in Advances in Computer Chess 2 (ed. M.R.B. Clarke), Edinburgh: Edinburgh University Press.
- Fine, R. (1941) Basic Chess Endings, McKay & Co., New York.
- Fisher, R.A. (1948) Statistical Methods for Research Workers. Edinburgh: Oliver and Boyd.
- Fisher, R.A. (1951) The Design of Experiments. Edinburgh: Oliver and Boyd.
- Halstead, M.H. (1977) Elements of software science. New York: Elsevier.
- Huberman, B.J. (1968) A Program to Play Chess Endgames. Ph.D. dissertation, Palo Alto: Stanford University.
- Knuth, D.E. (1976) Mathematics and computer science: Coping with finiteness. Science, 194, Research Memorandum 1235-1242.
- Michie, D. (1976) An advice taking system for computer chess. Computer Bulletin Ser. 2, No. 10, Dec., 12-14.
- Michie, D. (1977) Practical limits to computation. Research Memorandum MIP-R-116 Edinburgh: Machine Intelligence Research Unit, Edinburgh University.
- Michie, D. (1982) Experiments on the mechanization of game-learning: 2 -- rule-based learning and the human window. The Computer Journal, 25, No. 1, 105-13.
- Michie, D. (1982a) Game-playing programs and the conceptual interface. SIGART, No. 80, special issue on game playing, (ed. M.A. Bramer) New York: The Association for Computing Machinery.
- Miller, G.A. (1956) The magical number 7, plus or minus 2: some limits on our capacity for processing information. Psychological Review, 63 81-97.

- Niblett, T. (1982) Validation of machine-oriented strategies in chess endgames. Ph.d. Thesis, Edinburgh: Machine Intelligence Research Unit, Edinburgh University.
- Niblett, T. (1982) A provably correct strategy for king and pawn versus king. in Machine Intelligence 10 (eds. D. Michie and Y.H. Pao). Chichester: Ellis Horwood, and New York: Halsted Press (John Wiley, in press).
- Nievergelt, J. (1977) The information content of a chess position, and its implications for the chess-specific knowledge of chess players. SIGART Newsletter, 62, 13-15.
- Piaseski, L. (1977) An evaluation function for simple King and Pawn endings. M.Sc. thesis, Montreal: McGill University.
- Quinlan, J.R. (1979) Discovering rules by induction from large collections of examples. Expert Systems in the Micro-Electronic Age, pp. 168-201 (Ed. D. Michie) Edinburgh: Edinburgh University Press.
- Shannon, C.E. (1950) Programming a Computer for Playing Chess. Philosophical Mag., 41, 256-275.
- Shapiro, A. and Niblett, T. (1982) Automatic induction of classification rules for a chess endgame. in Advances in Computer Chess 3 (ed. M.R.B. Clarke) Oxford: Pergamon.
- Stroud, J.M. (1966) The fine structure of psychological time. Ann. of the N.Y. Acad., 623-631.
- Tan, S.T. (1972) Representation of knowledge for very simple pawn endings in chess. Research Memorandum MIP-R-98 School of Artificial Intelligence, Edinburgh University.
- Zdrahal, Z., Bratko, & Shapiro, A. (1981) Recognition of complex patterns using cellular arrays. The Computer Journal, 24, No.3, 263-270.
- Zuidema, C. (1974) Chess, How to program the exceptions? Afdeling Informatica, 1W21/74, Amsterdam: Stichting Mathematisch Centrum.

## REFERENCES FOR CHAPTER IX BY CASE STUDY

### Three Mile Island

- Dickson, D. Rasmussen vindicated? Nature , 282, p.221, 15 Nov. 1979.
- Douglas, J. California report pinpoints hazards in layout of Three Mile Island control room. Nature , 279, p.264, 7 Jun. 1979.
- Eytchison, R.M. Report of the Technical Assessment Task Force on: Control Room Design and Performance; Staff Reports to the President's Commission on The Accident at Three Mile Island, Vol. III, pp.175-203, Washington D.C., U.S. Gov't Print. Off., Oct., 1979.
- Kemeny, J.G. et. al. The Need For Change: The Legacy of TMI; Report of the President's Commission on The Accident At Three Mile Island, Washington D.C., Oct. 1979.
- Kemeny, J.G. An extremely small malfunction . . . and then something terrible happened. Dartmouth Alumni Magazine, 72, No. 4, pp. 30-37, Dec., 1979.
- Myers, J. and Schultzy, T., A Computer Based Design of a Power Plant Control Center, IEEE Transactions on Nuclear Science, Vol. NS-22, February 1975.
- Schwartz, J. Harrisburg: counting the cost. Nature, 278 pp. 589-90, 12 Apr., 1979.
- Taylor, P. Nuclear energy: how the odds are stacked against its opponents. Nature , 277, pp. 594-5, 22 Feb., 1979.
- Torrey, L. The week they almost lost Pennsylvania. New Scientist, 82, pp. 174-8, 19 Apr., 1979.
- Voysey, H. The importance of responsible programming (comment). Computerworld U.K. , p.27, 29 Oct., 1980.
- Weinberg, A. "Those who attack nuclear energy show a cynical denial of human ingenuity." Nature , 281, p. 335, 4 Oct., 1979.



### Three Mile Island Industry Studies

ART-77(2815)-1, Human Engineering of Nuclear Power Plant Control Rooms and Its Effects on Operator Performance. The Aerospace Corporation for the U.S. Nuclear Regulatory Commission, February, 1977.

EPRI NP-308-SY, Human Factors Review of Nuclear Power Plant Control Room Design. Lockheed Missiles and Space Company for Electric Power Research Institute, November, 1976.

Final Safety Analysis Report, Three Mile Island Nuclear Station Unit 2. Metropolitan Edison, Jersey Central Power and Light and Pennsylvania Electric Company, April, 1974.

IEEE Std. 566-1977, IEEE Recommended Practice for the Design of Displays and Control Facilities for Central Control Rooms of Nuclear Power Generating Stations. Nuclear Power Engineering Committee of the IEEE Engineering Committee, 1977.

NUREG 76-6503 (SAND 76-0324), Preliminary Human Factors Analysis of Zion Nuclear Power Plant. Sandia Laboratories, October, 1975.

### Air Traffic Control

Brody, J. Pilot' Role in Accidents Draws Increasing Attention. in Research. The New York Times (last of series) pp.1,A23., Monday, 23 June 1980.

Michie, D. New Face of AI. Experimental Programming Reports: No.33, Machine Intelligence Research Unit, University of Edinburgh. Oct., 1977. (also Hoogovens hot strip mill).

Torrey, L. Near Misses in the sky. New Scientist , 87, No. 1210, pp. 188-191, 17 July 1980.

Witkin, R. Near-Collisions Persist in the Sky and on Runways. The New York Times , (4th of 5 articles on aviation safety) pp.1,26. Sunday, 22 June, 1980.

Witkin, R. Panel Offers Plan to Bolster F.A.A. Technical

- Expertise. The New York Times , p.A16, Friday, 27 June, 1980.
- Witkin, R. Jet Pilot Tells of Close Traffic 10 Miles From Kennedy. The New York Times , Thursday, 10 July, 1980.
- Voysey, H. Problems of mingling men and machines. New Scientist , 75, No. 1065, pp. 416-417, 18 Aug. 1977. (also Hoogovens hot strip mill)
- NORAD Military Computer
- Cross, D. Computer is shut down after second missile alert. The Times , p.1 , Sunday, 8 June, 1980.
- Halloran, R Computer Error Falsely Indicates A Soviet Attack. The New York Times , p.A14, Friday, 6 June, 1980.
- Halloran, R. Brown Says False Alarm Cannot Activate Missiles. The New York Times , p.A16, 10 June, 1980.
- Sulzberger Jr., A.O Error Alerts U.S. Forces to a False Missile Attack. The New York Times , p30. , Sunday, 11 Nov., 1979.
- Torrey, L. The computer that keeps on crying wolf. New Scientist , pp.375-6, 26 June, 1980.
- Wright, P. 'Alert' mistake difficult to resolve. The Times, Monday, 9 June, 1980.
- Wright, P. University seeks improved computer reliability. The Times , p.4e , Wednesday, 11 June, 1980.

Royal Dutch Steel

Crawley, J.E.; Ketteringham, P.; Vine, D.; and Price, C.;

Cover and Summary of Project Report: Human Factors Evaluation Hoogovens No.2 Hot Strip Mill, British Steel Corporation, Battersea Laboratory, FR/251/76(B), August, 1976.

Community Ergonomics Action; Human Factors Evaluation at Hoogovens No. 2 Hot Strip Mill; Proceedings of the Meeting on 26-27 October 1976 in IJmuiden/Amsterdam.

(See also references for Air Traffic Control)

## Appendix A.1

Correction of Type 0, Type 1 and Type 2 Harris Program Errors.

Of the 37 Type 0, Type 1 and Type 2 errors, only Type 0 represent genuine Harris program errors. Type 2 errors can be trivially corrected by marking all values in the database which evenly divide 27 as meaning Black-to-move draws, i.e.  $459 = 27 \times 17$ , remainder 0, means Black-to-move draws. These were marked incorrectly in procedure BASVAL as wins. Type 1 errors required some tracing to discover why IRESB=4 (meaning Black-to-move draws, according to the database and procedure BASVAL) rather than IRESB=3, meaning that Black-to-move loses. After some careful tracing of the values passed by procedure BASVAL and its subsequent decomposition of database values, Type 1 errors were removed simply by replacing a division sign (/) with the Algol function for integer division (div). It seemed type 0 errors, though genuine, could also be trivially corrected by:

- (a) not allowing the program to "imagine" all positions where WPRANK=2 as if WPRANK=3 for convenience or
- (b) handling the case where WP:a2, BK:a1, as a special one, although there will be other positions where WPRANK cannot be "imagined" as 3 correctly.

In order to secure as much generality as possible, approach (a) was chosen to correct Type 0 errors. At first the following was added to the test IF WPRANK=1 : AND NOT(BKLIST[WPLOC]) AND NOT(ADJACENT(WK,WP)) THEN WPLOC:=WPLOC + 1. This means, if the WP is on rank 2 counting from 1, (Harris uses a board representation which numbers ranks and files from 0 to 7) and the BK cannot immediately move to the square the WP is on, nor is the WK adjacent to the Pawn, then "imagine" the WP on rank 3. This change successfully corrected Type 0 errors. However, in retesting seg-

ments of the sample 2002 positions, a new error, due to the above correction, was discovered. That is, positions such as WP:a2, WK:a1, BK:c1, where the WK is adjacent to the WP and the WP is not on the BK's list of possible moves, do require that the Pawn's rank be incremented in order for them to be evaluated correctly as wins with Black-to-move. Finally, the following clear, but very inefficient, code was added to the test IF WPRANK=1 to remove Type 0 errors: AND BMOVE AND (BK-WP+FLANK[1] OR BK-WP+FLANK[2] OR BK-WP+DOWN) THEN WPLOC:=-WLOC; ELSE WPLOC:=-WPLOC+1.

This means, if it is Black-to-move and the BK is anywhere just behind the WP, it is incorrect to "imagine" the WP on rank 3 instead of rank 2.

A further test on another sample of 2000 positions (Nos. 1001-2000 see tabulation on p.31 ) resulted in only one error, or 99.95% correctness. This error, with WK:b4, BK:b2, WP:a2, can also be classed as a Type 0 error. This subset of Type 0 errors must generalize across all 4 files the pawn is evaluated on. The addition of the condition: OR WK-WP + AHEAD[1] OR WK-WP + AHEAD[2] OR WK-WP + AHEAD[3] to the earlier mentioned amendments, cured this problem.

## Appendix A.2

### Correction of Type 3 and Type 0.n Harris Program errors.

Due to the biased nature of Type 0.n and Type 3 errors, some "easy fix" to a programming error was sought. The error had to be related to some faulty increment of the WPRANK which did not affect the values of positions where White-to-move could win by advancing the pawn, but would have a vital effect on positions where Black had only one drawing move. As predicted, the error was quite obvious after reviewing the changes described in APPENDIX A.1; ELSE WPLQC:=WPLQC + 1 should have been: ELSE IF WPRANK=1 THEN WPLQC:=WPLQC + 1. Finally this implies that in most positions with the pawn on the second rank it is safe to "imagine" the pawn on the third rank, but in some cases it is absolutely wrong. Therefore most of the reported errors for the sample of 800 randomly selected positions were caused by this mistake.

Type 3 errors (i.e. WP:c3,WK:h3,BK:h7, B to move) were removed by the discovery and correction of a small encoding bug in PROCEDURE RANK234 (see fig. 5), where the intermediate test performed for special distant King relationships, IF WKIN >= DIST(BK,WP+4) was changed to IF (WKRANK <= WPRANK AND (DIST(WK,WK+UP) >= DIST(BK,WP+4) -1) AND NOT(SSOP OR KOPOP)). The PROCEDURES SSOP and KOPOP are described in Section 2.1.3 and they were intended to ensure that positions where the WK could get the opposition with his pawn on rank 3 or 4 were evaluated as wins only when the King could get the opposition ahead of the pawn. Thus the errors described in Bitner & Hansche's report for their figs. 6 and 7 and similar positions, were removed.

APPENDIX B: KPK MANUAL

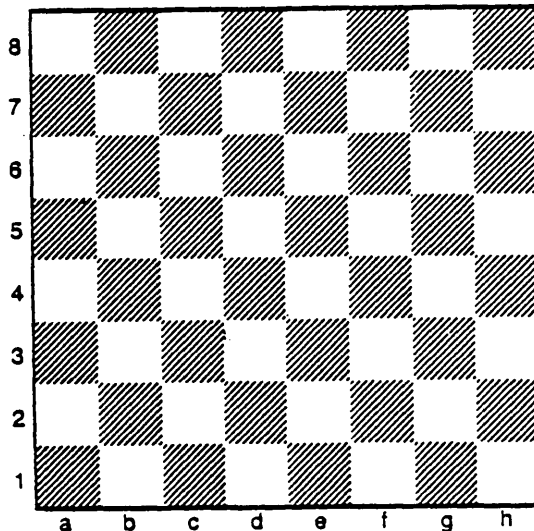
CONTENTS

- I. Describe Legal Moves (Rules)
- II. Examples
- III. Objectives of Play

I. Rules

What follows is a description of the rules of play for a very specialized yet non-trivial subset of the game of chess. The movements of the 3 pieces involved, a White King (WK), a White Pawn (WP), and a Black King (BK), are very limited, though one mistake can critically alter the result. The possible results are either White wins by obtaining a position where his pawn will be decisive, or the position is a draw.

Chess is played on an 8 x 8 board whose squares are marked as in the following diagram. The board is labelled from White's point of view.



Pawns can only move forward one square at a time on a file (column) except from their starting position on rank (row) 2, when there is the option of advancing them one or two ranks.

For the purposes of this miniature game of chess, if the WP can safely reach the 8th rank, then White is said to have a won position. Kings can move one square at a time in any direction (forward, backward, left, right, or diagonally). Kings cannot move adjacent to each other. The BK cannot move to a square diagonally ahead of the WP. If White advances the pawn to a square

which is diagonally directly below the BK, then Black is said to be in "CHECK". White and Black alternate moves; A move is denoted by piece to square, i.e. 1.Kd7 means that White's first move is WK to d7. Black moves are indicated by "... " before the move played, i.e. 1. ...Ka7 indicates that Black's first move is BK to a7. Pawn moves are denoted simply by the square the WP is to be moved to, i.e. 1.c5 is WP from c4 to c5.

## II. Examples

Fig 1: WK:d4, BK:d6, WP:c2, W to play.

Here White's legal moves are Kc4, Ke4, Kc3, Kd3, Ke3 (but not Kc5, Kd5 or Ke5 since these moves would bring the Kings adjacent to each other) and the two pawn move options c3 or c4.

Fig. 2: WK:d3, BK:d6, WP:e4, W to play.

Here White can play Kc4, Kd4, Kc3, Ke3, Kc2, Kd2, Ke2, and e5+ (Pawn to e5, Check). On this last move Black could capture the Pawn by ... KxP by simply replacing the Pawn with his King on the square e5 and removing the Pawn from the board.

Fig. 3: WK:e5, BK:c6, WP:d4, B to play.

Here Black can play ... Kb7, Kc7, Kd7, Kb6, Kb5, but not ... Kc5 (since this would be moving into check from the Pawn) nor ... Kd6 or ... Kd5.

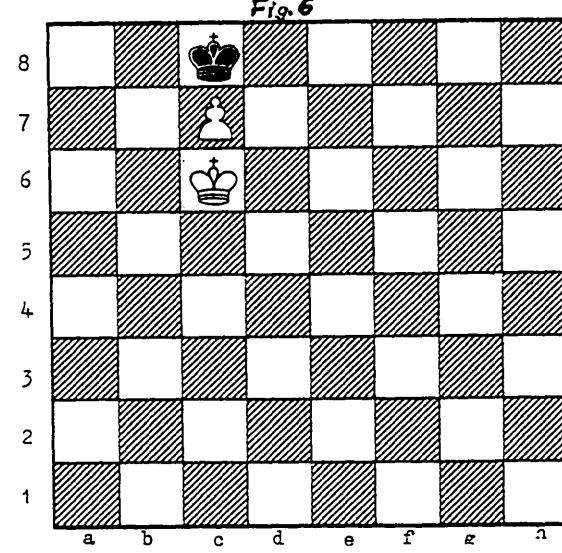
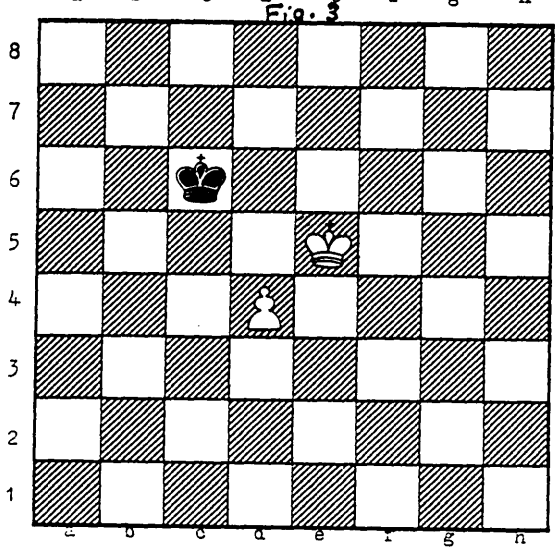
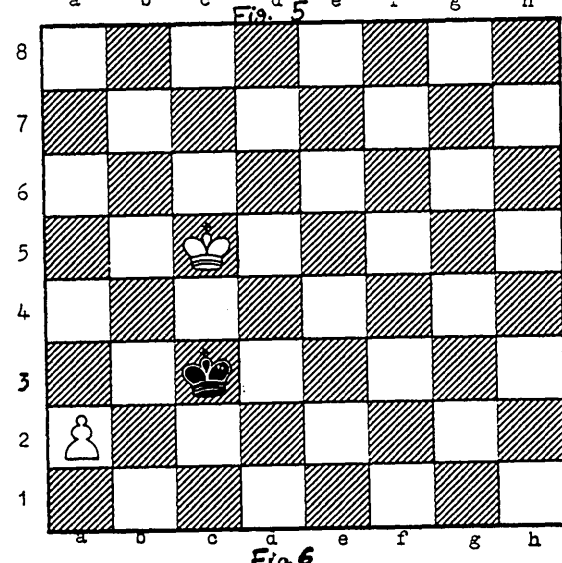
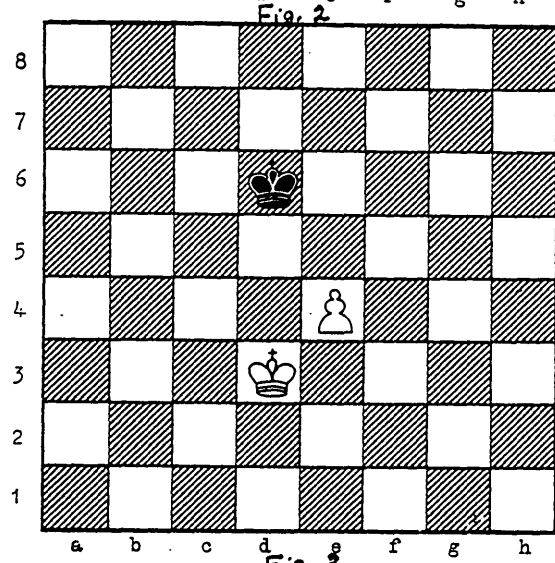
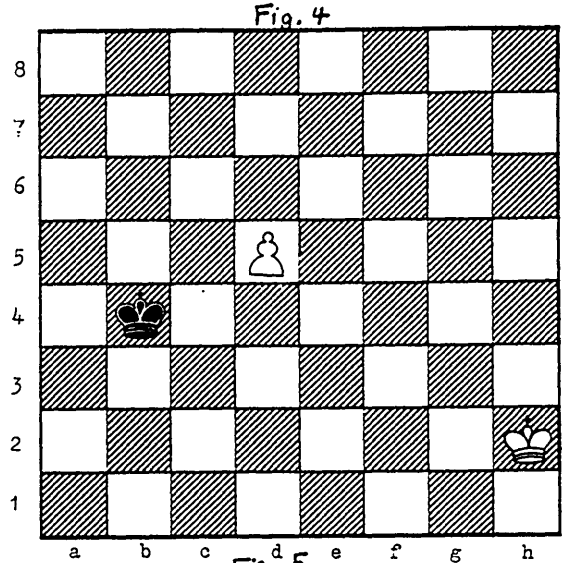
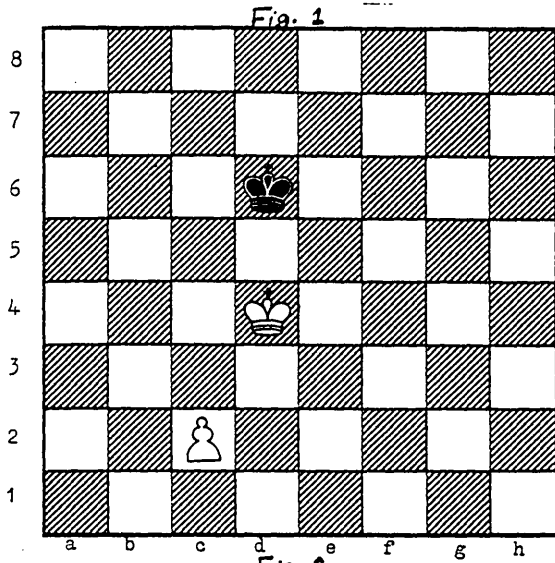
Fig. 4: WK:h2, BK:b4, WP:d5.

Here with Black to play 1. ... Kc5 is possible and Black captures the Pawn next move (draw). However with White to play 1.d6 is possible and the Black King cannot catch the Pawn and prevent it from reaching d8 safely.

Fig. 5: WK:c5, BK:c3, WP:a2.

An unusual position whereby the Black King appears close enough to the Pawn to capture it, but because of special features is unable to do so: ... Kb4, ... Kc4, and ... Kd4 are all illegal because each of these moves would bring the Kings adjacent to each other. The move ... Kb3 is illegal because it moves into check from the Pawn and on ... Kb2 White would play 2.a4 and then the Pawn would be able to 'run away' and reach the eighth rank.





### III. Objectives

White tries to win by gaining access to the 8th rank for his Pawn. This is achieved either by simply advancing the Pawn and "beating" the Black King to that square or by the White King's support of the Pawn by coaxing the BK from that square, or by a combination of the above methods. Black draws by either (1) Capturing the Pawn or (2) being stalemated or obtaining a position which ultimately leads to stalemate.

Stalemate occurs when it is Black's turn to move, he is not in check, but cannot make any legal moves. Fig. 6 illustrates a Stalemate.

APPENDIX C: Performances of 8 Control Subjects on KPK Quiz

NAME	RATING	R	RR	RW	WW	WR	PERCENT RIGHT (100 x R/24)
1. K. Sorn	1315	12	3	3	3	0	50.0
2. G. Morrison	1485	18	7	1	1	0	75.0
3. J. Harmon	1606	17	6	1	1	1	70.8
4. B. Potter	1700	18	7	1	1	0	75.0
5. P. Anderson	1703	16	5	2	1	1	66.7
6. T. Dougherty	1740	20	6	1	2	0	83.3
7. D. Moy	1430	13	4	1	4	0	54.5
8. J. Cook	1570	19	6	2	1	0	79.2
AVGS.:	1569	16.63	5.50	1.5	1.75	0.25	69.3

Key

- R = No. of correct responses of maximum 24.
- RR = No. right responses for right reasons, i.e. correct "best" moves of maximum 9.
- RW = Right Value, Wrong Best move.
- WW = Wrong Value, Wrong Best move.

Note: RR + RW + WR + WW = 9.

## Appendix D

Tests with a small sample of rated human chessplayers

<u>Name</u>	<u>Age</u>	<u>Rating</u>	<u>Pre-Test</u>	<u>Post-Test</u>
1. C. Tomlinson	11	1175	6	9
2. A. Condie+	16	1570	10	9
3. A. Hunter	18	1905	10	10
4. D. Holmes	16	1930	10	10
5. A. Wright	15	1960	10	10
6. A. Biancini	16	1975	10	10

---

+ female subject

This testing with a small sample of subjects using the Pre-Test and Post-Test administered with the Niblett Advice Text (open and closed-book) gives some indication that ratings of at least 1600 would be required for humans to correctly evaluate KPK positions.

## Appendix E: Notes

1. As reported on page 32 , it was later discovered that only Black-to-move (B-T-M) positions were being compared for correctness with database values. Hence all figures of percentage correctness which follow refer only to B-T-M positions unless stated otherwise.
2. Since Beal's Fortran program was found to be 100% correct when tested against Clarke's database, it is clear that these were transcription errors.

PUBLISHED PAPERS

Recent Developments in Computer Chess

by

D. Kopec<sup>†</sup>  
Machine Intelligence Research Unit  
University of Edinburgh  
Scotland, U.K.

"If one could devise a successful chess machine one would seem to have penetrated to the core of human intellectual endeavour."

A. Newell, J. C. Shaw and H. A. Simon (1963)  
"Chess playing programs" in Computers and Thought

The above was written fourteen years ago and since that time it must be admitted that progress has been disappointing.\* However some recent developments are encouraging:

- (1) a. Northwestern's Chess 4.5 won the Class B section (USCF rating scale 1600-1800) of the Paul Masson Tournament, held in July in California (5 - 0).
- b. Chess 4.5 also defeated a Class A player in an individual match game (see page 41 for game record).
- (2) Special chess hardware with a 10-ply lookahead has been designed by Greenblatt and Missouris. Two Experts have been included among its victims. It can look at more than 100,000 boards/sec.

Even in the end-game, generally agreed to be the most difficult phase of chess, there have been some successes. In the Soviet Union, the entire space of all R + P vs. R positions has been computed out and stored as a lookup table. Many masters and grandmasters have been known to go wrong in such endings. The same was done with Q + P vs. Q. David Bronstein, a former World Championship contender, consulted that database for the correct strategy to win an adjourned game.

The following notes concern interesting features of two chess programs which I have myself been concerned with. Not all the ideas were implemented.

Work at Dartmouth College, New Hampshire, USA

The computer chess project at Dartmouth is headed by Dr. L. Harris. Workers have included W. Montgomery, H. Terrie, D. Levner and myself, among others. The program was written in GCOS, the assembly language for the Honeywell 635. Dartmouth's program was the first program to challenge Northwestern's

---

<sup>†</sup>U.S. National Master

\*This article was written before CHESS 4.5's later successes.

ascendancy in the ACM tournaments. In the 1973 ACM U.S. Computer Chess Championship, held in Atlanta, Northwestern's program was lucky to draw against Dartmouth. It only did so because the latter had no repetition check. Particularly encouraging was the fact that in a game of more than 50 moves, Dartmouth was never in a losing position.

The program is divided into two major evaluation functions,  $\hat{h}$  and  $\hat{d}$ .  $\hat{h}$  is concerned with the "soft", positional features of a given board position, while  $\hat{d}$  is concerned with the "hard" tactical features of a position. The specific chess concepts which comprise  $\hat{d}$  and  $\hat{h}$  are called "Detectors". A set of related detectors are assigned various values (weights) and are put into a table.  $\hat{h}$  includes tables such as Centre Control, Piece Mobility, Pawn Structure, King Safety, etc., while  $\hat{d}$  includes tables such as Pins, Forks, Discovered Attacks, Levers. The program is also divided into modules (Opening, Middle Game, and Endings) which allow greater flexibility in the assignment of weights. For example, in the opening, Piece-development, Centre control, and King safety are stressed. A persistent problem which many programs still have is the too early development of the queen, because of its tremendous square control, mobility, and ability to produce threats. By assigning a value of -300 (where 100 = pawn) to every minor piece (B or N) still on the back rank, piece development is given prominence, since the program tries to get rid of these initial negative values. Other examples of tables which employ modular flexibility are Occupation of the Centre, and Rook on 7th. Greater weights are assigned to these in the middle game and ending than in the opening, to avoid moving the same pieces too often, before others have moved at all.

An idea which was never fully implemented was that of an "Attack-Defence Ratio". This is a measure of the difference between the sum of the forces attacking the quarter of the board where the enemy king is located and the sum of those forces which defend the same squares. If this difference in force is greater than a certain threshold value, an "alarm" is set off which results in a higher  $\hat{d}$  value and an increase in the depth of search. In this manner, long sacrificial variations are more carefully investigated. A benchmark of sacrificial positions would be a good test for its effectiveness.

Dartmouth's most "informed" table was the one on pawn formations, called "PFORM". Among its standard detectors were Isolated Pawns, Backward Pawns, Doubled Pawns, Passed Pawns, and Duos. Detectors such as Chains, Mini-chains, Shielded Backward Pawns, Potential Passed Pawns, and the table, "Levers", were among the more esoteric concepts which were added later. Many of these definitions were taken directly from Hans Knoch's classic work Pawn Power (see Knoch, 1959). The concept, Levers, using a modified definition of my own--"pawn moves which improve our formation and hurt our opponent's"--proved useful in the recognition of critical pawn moves. In addition, the levers concept helps to guide the placement of pieces especially in the opening and middle game. It could also help toward plan formation. Some further pawn formational concepts from Pawn Power which were never programmed were Outposts and Weak Square Complexes. The Dartmouth program is probably, in theory, capable of more sophisticated pawn formational evaluations than any other program; however their implementation is rudimentary. The program had at one time approximately 50 detectors in various tables and many others were planned.



Another idea due to Harris was that of "Heuristic Packets" for specific openings, by which important goals (inherent in the openings), could be stored as pieces of information. These Heuristic Packets were to include such information as the importance of controlling a certain square, an important lever, a square to aim a piece for, etc. and record kept of successes and failures which could then be used to modify the evaluation parameters. Unfortunately, this idea was not properly implemented.

Work at the Machine Intelligence Research Unit, Edinburgh

Tan's program (see Tan, 1977) breaks down any K + P ending into a basic description of its components. Using a vocabulary which is defined in Pawn Power, the pawn formations are broken down into Fronts, the relationships between opposing pawns. The role and relation between each pawn within these Fronts and Groups (islands of friendly pawns) is then defined, as shown in the Table.

Table of Pawn Relations

	name	graphical notation	code
hostile relations	counter pawn	<->	1
	ram	<+>	2
	sentry	<..>	3
	lever	<.+.>	4
friendly relations	duo	=	5
	twin	= x	6 (inverse:7)
	potential protector	= >	8 (inverse:9)
	protector	= / = >	10 (inverse:11)

An enemy pawn ahead on the same file is a counterpawn, and a sentry when it is on a neighboring file....Counterpawns and mutual sentries of distance 1 are called rams and levers respectively. Friendly relations give rise to a duo when the pawns are abreast on two neighboring files, and a twin (doublepawn) when they are on the same file. A backward neighbor is a protector (distance 1) or a potential protector (distance > 1)."

From S. T. Tan (1977)

Some of these relations may be very useful if developed further. For example, if a pawn is "over-loaded", in that it is performing several roles at once, its removal may lead us to a winning strategy. An example in practice is shown by the problem position in Figure 1, which I have taken from Basic Chess Endings. Its solution becomes clearer when we consider the number of roles performed by the pawn on square 33 (square = 9\*File + Rank).

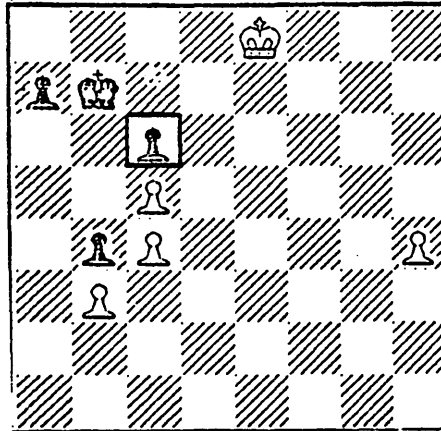


Figure 1: From Basic Chess Endings (R. Fine)

Thus, 33[1,31], i.e. pawn on 33 is a Counterpawn to pawn on 31  
33[2,32], i.e. pawn on 33 is a Ram to pawn on 32  
33[3,21], i.e. pawn on 33 is a Sentry to pawn on 21  
33[8,22], i.e. pawn on 33 is a Potential Protector to pawn  
on 22.

(See preceding Table for definitions.)

Rather than queening the passed RP immediately, which would result in a quick draw due to Stalemate, the solution lies in White's capture of that pawn on 33.

A graph representation for the same position is given below, as Figure 2, using Tan's notation as in Table 1, with the addition of "Λ" to denote a passer (my notation).

Another concept put forward by Tan is that of an ADD (Attack Defense Diagram). (See Figure 3.) Some of the evaluations suggested for an ADD are as follows: (1) Relations within fronts; (2) Defenses to threats of (1); (3) Possible attacks of Kings against Pawns; (4) Defenses to (3); (5) Support possibilities; (6) Joint attacks. Tan also breaks down nearly all possible relationships that might go into the ADD into B.N.F. (Backus-Naur Form). As yet there is no working program for the ADD. As a step towards designing one I have tested how the evaluations would work on some simple K + P endings. It seems that at least three concepts must be added: (1) Opposition, (2) Triangulation, (3) Outside Passed Pawns. These features are important, and occur frequently.

<u>GRAPHICAL NOTATION</u>	
<b>HOSTILE RELATIONS</b>	Counterpawn
	Ram
	Sentry
	Lever
<b>FRIENDLY RELATIONS</b>	Duo
	Twin
	Potential Protector
	Protector
	Passer (my notation)

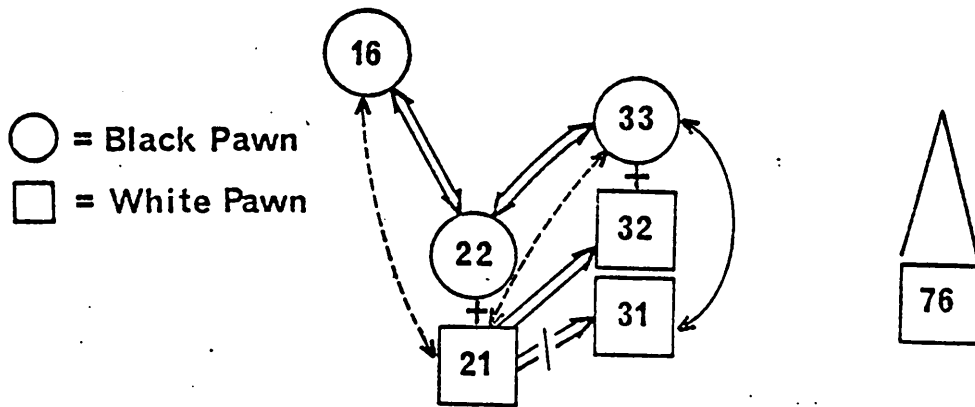
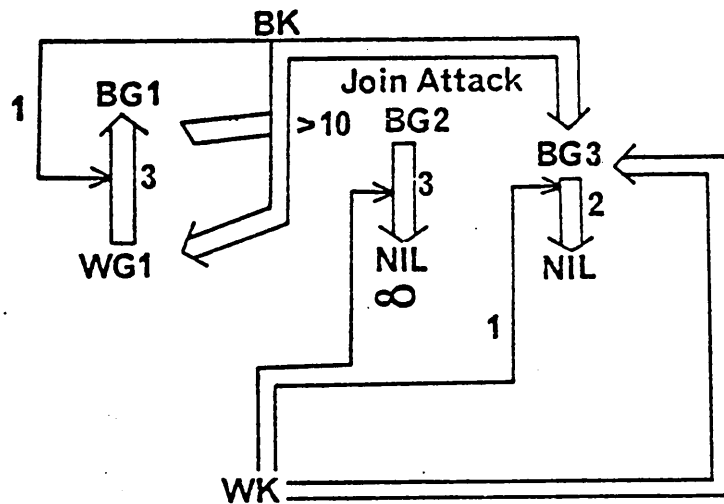


Figure 2: Graphical representation for chess position shown in Fig. 1

TAN'S ATTACK DEFENCE DIAGRAM



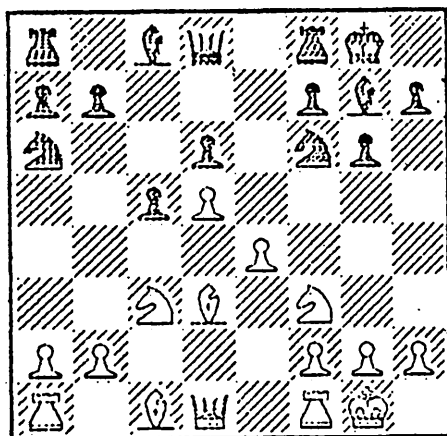
Symbols: Threats  
 Defenses

Figure 3: Same position, in the form of an Attack-Defence Diagram

Common Faults of Existing Programs

One obvious fault is due to what Berliner has termed "the Horizon effect". That is to say, given a fixed depth of lookahead, and the usual definition of quiescence (no checks or captures) a program may waste time and material in order to defer a loss which to a deeper search can be seen as inevitable; or, alternatively it grabs shortsightedly at a small gain, oblivious to a major one beyond its horizon. It might be helpful to consider all material gain threats in the same league as captures and checks. In other words, a position is quiescent when there are no checks, captures, or threats to gain material. Another way to deal with this problem might be with a one-ply search beyond all double threats. However, a precise definition of a double threat is not easy for the programmer.

Other program faults are due to its lack of knowledge. We cannot expect too much from computer chess play when there is so little to guide its decision-making process. The human chess player, particularly at the master level, employs rapid knowledge-based pattern recognition or feature detection. There are also many experiences (good or bad) which cause him to revise his thinking. Such experiences lead to the formulation of a hierarchy of rules which can guide play. Consider the following position which might easily occur in the Benoni Defense.



Black has just played N-QR3.

Most computers would probably play B X N here, if out of their opening library. On pawn structural grounds alone the move is very reasonable (Black's isolated doubled Queen's Rook Pawn). However experience has shown that Black's resulting two Bishops and half-open Queen's Knight file more than compensate for the weak-looking pawns. Another piece of knowledge which most programs lack is when the qualitative value of pieces changes. When is a Bishop worth more than a Rook? When is a Knight worth more than a Bishop? Many such conceptualizations over a chessboard are based on the style and originality of individual players.

### Representation of Human Chess Knowledge

I have only recently been convinced by Professor Donald Michie's argument that every "bit" of chess information assimilated by Bobby Fischer over 34 years of life can in principle be input into a computer. Thus, if we consider that the fastest rate of human information input is 30-50 bits/sec. and multiply this by the seconds in a year and then by 34 (Fischer's age) we still have a quantity of information which can easily be stored and retrieved by a computer. Therefore the problem is one of representation, not of space.

When we consider that at most a chess master will "look at" (search) 50 boards versus the more than 100,000/sec. which can be viewed by Greenblatt's new chess machine, Cheops, we see the need for some rules of compression of human chess knowledge. Such rules as: two bishops are an advantage, knights are better than bishops in closed positions, don't block the Queen's Bishop Pawn with the Queen's Knight in Queen's Pawn openings, and the continuous refinement of such rules by human players must surely strengthen any computer chess program. But how many of the rules by which masters play we should aim to use in a program is another question. The number has been estimated by two different sources, Simon and Gilmartin (1973) and Nievergelt (1977) as lying in the range of 10,000 to 100,000.

### Challenge to Reader

As a tail piece, I wish to put forward an hypothesis that a chess game between two computers can easily be distinguished from one between Bobby Fischer and a master. It is based on the general statements about computer play made in this paper and also the following points:

- (1) Integration of Position (Entropy)
- (2) Development of Pieces
- (3) Material Equality
- (4) Simplification.

I suggest the reader take a few minutes to look at each of the positions in Figures 4-9, and then decide for each whether it is between two computers, or Fischer vs. a Master. Each of the positions occurred after 20 moves of play in the game. The solutions are on page 46.



## Bibliography

- Berliner, H. J. (1974) Chess as Problem Solving: The Development of a Tactics Analyzer. Ph.D. Dissertation, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA.
- Fine, R. (1941) Basic Chess Endings. Philadelphia: David McKay; London: Bell.
- Harris, R. L. The Heuristic Search and the Game of Chess - A Study of Quiescence, Sacrifices and Plan Oriented Play. In Proceedings of IJCAI 1975, Vol. 1, Tbilisi, Georgia, USSR. pp. 334-339.
- Knoch, H. (1959) Pawn Power. New York: David McKay Company, Inc.
- Levy, D.N.L. (1975) The 1975 U.S. Computer Chess Championship. Computer Science Press, Inc., Woodland Hills.
- Newell, A., Shaw, J. C. and Simon, H. A. (1963) Chess playing programs. In Computers and Thought (eds. E. A. Feigenbaum and J. Feldman) pp. 39-70. New York: McGraw-Hill.
- Nievergelt, J. The information content of a chess position, and its implication for the chess specific knowledge of chess players. SIGART Newsletter (in press).
- Simon, H. A. and Gilmartin, K. (1973) Cognitive Psychology 5. 29-46.
- Tan, S. T. (1977) Describing Pawn Structures. In Advances in Computer Chess 1 (ed. M. Clarke) pp. 74-88. Edinburgh: Edinburgh University Press.
- Wade, R. G. and O'Connell, K. J. (1972) Bobby Fischer's Chess Games. Garden City, New York: Doubleday & Company, Inc.

-ooOoo-

### Answers to "Challenge to reader", by D. Kopec, page 34

- Fig. 5. IRON-FISH vs. CHUTE 1.2.  
Fig. 6. Fischer-Nicevski, Zagreb 1970.  
Fig. 7. Fischer-Hamaan, Natanya 1968.  
Fig. 8. SORTIE-CHESS 4.4.  
Fig. 9. DUCHESS-IRON-FISH.  
Fig. 10. Fischer-Camera, Siegen 1970.

(All computer positions from U.S. Computer Championship 1975 by D.N.L. Levy)

Note that in all the Fischer games, at move 20, as opposed to the computer games, there is more integration of forces. The lines between black and white forces are more clearly drawn. Fischer games have more development, i.e. fewer pieces on the back rank, and material is more even. In general, computer games show more simplification at move 20. I base this last point on the idea that since current programs don't know what to do with their pieces, there will be more trades.

# Pattern-based representation of chess end-game knowledge

I. Bratko\*, D. Kopec and D. Michie

Machine Intelligence Research Unit, University of Edinburgh, Hope Park Square, Meadow Lane, Edinburgh EH8 9NW

Students of computer chess aim at an operational theory of Master skill—operational in the sense that it can be run on the machine. In one form of the aspiration Masters must be defeated across the board under full tournament conditions, so far achieved only for 'blitz' chess but not for play under standard time control (see *SIGART Newsletter* No. 62, 1977). Another form of the 'Master skill' aspiration aims at *correct play* for defined subsets of chess. It is not known whether 'strong mastery' in this sense is attainable for the complete game or whether chess is 'hard' in the sense of Knuth (1976). We can, however, start with elementary endings such as King and Queen *versus* King (denoted KQK), KRK, KBBK, KBNK and KPK, and seek to extend mastery backwards a step at a time into the game's increasingly complex hinterland.

Work at Edinburgh follows the second approach, seen as a means for studying forms of knowledge representation in relation to three *desiderata*: (a) forms more powerful than present programming languages for specifying strategies, (b) forms more suitable for proofs of correctness of strategies and (c) forms more convenient for automatic optimisation of strategies ('machine learning').

None of the above listed end-games contains anything problematical from a Master's point of view and computer programs embodying correct strategies have been written for all of them. In reviewing this work Bramer (1977) remarks that the task, not of playing such end-games correctly, but of expressing in program form the knowledge required for correct play, has turned out to be surprisingly and disproportionately hard. For his own implementations of KRK and KPK Bramer uses pattern-based models of a general kind now accepted as indispensable to the extension of machine mastery into more complex chess subdomains. Such exercises as KRK and KPK can be done (with some difficulty) *without* special programming tools; but it needs only a small step in the direction of greater complexity to bring us into territory where the use of such tools become critical.

In this paper we describe pattern descriptorial aids to strategy building in two areas more complex than KRK, KPK, KQK and the rest; namely (a) pawns-only positions and (b) the defence of king and knight against king and rook.

## Describing pawn structures

Tan (1977) has developed a program which breaks down any K + P ending into a basic description of its components. Using a vocabulary which is defined in Kmoch's (1959) *Pawn Power*, the pawn formations are broken down into Fronts, i.e. islands containing opposing pawns, and further subdivided into Groups (same colour only). The rôle of each pawn in terms of its relationships to other pawns is then defined, as shown in the upper part of Fig. 2 which uses terms explained by Tan as follows:

'An enemy pawn ahead on the same file is a *counterpaw*, and a *sentry* when it is on a neighbouring file . . . Counter-pawns and mutual sentries of distance 1 are called *rams* and *levers* respectively. Friendly relations give rise to a *duo* when the pawns are abreast on two neighbouring files, and a *twin* (doublepawn) when they are on the same

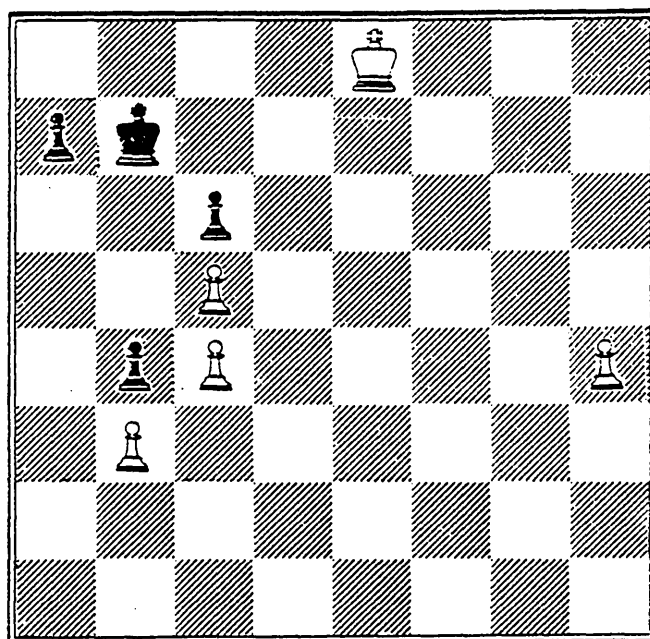


Fig. 1 From *Basic Chess Endings* (R. Fine)

file. A backward neighbour is a *protector* (distance 1) or a *potential protector* (distance > 1).'

Some of these relations may be very useful if developed further. For example, if a pawn is 'overloaded', in that it is performing several rôles at once, its removal may lead us to a winning strategy. An example in practice is shown by the problem position in Fig. 1 taken from Fine's (1964) *Basic Chess Endings*. Its solution becomes clearer when we consider the number of rôles performed by the pawn on square 33 (square = 9 × file + rank).

Thus,

33 [1, 31], i.e. pawn on 33 is a Counterpaw to pawn on 31,

33 [2, 32], i.e. pawn on 33 is a Ram to pawn on 32.

33 [3, 21], i.e. pawn on 33 is a Sentry to pawn on 21.

33 [8, 33], i.e. pawn on 33 is a Potential Protector to pawn on 22.

Rather than queening the passed RP immediately, which would result in a quick draw due to stalemate, the solution lies in White's capture of that pawn on 33.

A graph representation of the same position is given in the lower part of Fig. 2, using Tan's notation with the addition of  $\wedge$  to denote a passed pawn.

Another concept put forward by Tan is the ADD (Attack Defence Diagram, see Fig. 3). Some of the relations proposed for an ADD are as follows: (a) relations within fronts; (b) defences to threats arising from (a); (c) possible attacks of kings against pawns; (d) defences to (c); (e) support possibilities; (f) joint attacks. Tan also breaks down the relationships that go into the ADD into Backus-Naur Form. As yet there is

\*Present address: Jozef Stefan Institute, Ljubljana, Yugoslavia.



		GRAPHICAL NOTATION	
HOSTILE RELATIONS	Counterpawn	←→	
	Ram	< + >	
	Sentry	←- - ->	
	Lever	← + →	
FRIENDLY RELATIONS	Duo	==	
	Twin	==X	
	Potential Protector	==>	
	Protector	==/>	
	Passer (our notation)	>	

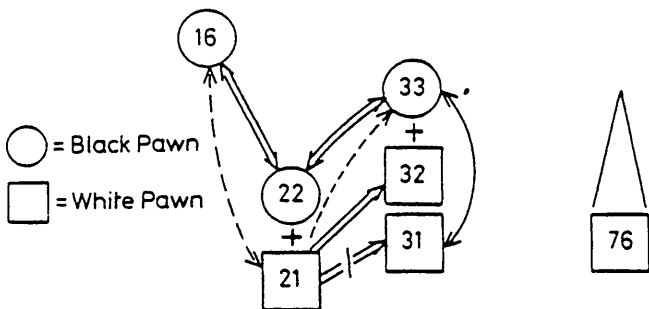


Fig. 2 Graphical representation for chess position shown in Fig. 1

no working program for the ADD. As a step towards implementation, we have tested how the evaluations would work on some simple K + P endings. It seems that at least three concepts must be added: (a) opposition, (b) triangulation, (c) outside passed pawns. These features are important and occur frequently.

#### AL1 'advice taker' system

In a related use of pattern-based representations of chess knowledge we have developed a linguistic vehicle for applying McCarthy's (1959) 'advice taker' concept. In Advice Language 1 (Michie, 1976) knowledge is conveyed to the system in the form of one or more *advice tables*, each specifying a number of *rules*. A rule is applied to a position (in a manner familiar to commercial users of decision tables, and to academic users of production systems) if and only if the position matches the rule's 'condition pattern'. Associated with each rule is a list of *pieces of advice*. Each piece of advice is specified in terms of Huberman-type (Huberman, 1968) better-goals and holding-goals, together with move-constraints to control the branching of the search and a depth limit to terminate it when no way has been found of achieving the given better-goals.

AL1 has been implemented in the POP-2 programming language as a package consisting of four comparatively independent modules (core-occupancy of compiled code on the PDP-10 is shown in parentheses):

1. A *problem-solver* performs tree search in whatever problem space is specified to it by the legal move generator, using the domain specific knowledge contained in the currently loaded Advice Tables (2K).
2. An *Advice Table editor* acts as a link between the system and the user, enabling him to create, extend and modify the system's Table held knowledge interactively (4K).
3. A *playing module* executes a strategy generated by the problem-solver in the form of a Huberman-type forcing tree (6K).
4. *Chess-relevant but subdomain independent POP-2 predicates* act as the building blocks from which the table writer

assembles relevant patterns and pieces of advice from which to construct his rules (13K).

In addition subdomain specific predicates are normally required for each new Advice Table. The POP-2 system itself occupies 19K 36 bit words of store.

Modules 1 and 2 are chess independent and can be used for solving other combinatorial problems. The domain specific knowledge directs the action of module 1 in the following way: a rule is invoked by a pattern-match with the current situation (chess position) and the corresponding pieces of advice are then tried one by one until module 1 can find a 'forcing tree' that guarantees the achievement of better-goals while preserving holding-goals.

Considered as an ultra-high level programming language, AL1 seems to provide a natural means of describing heuristics in combinatorial algorithms. In one experiment (Michie, 1976) the King + Rook v. King ending (KRK), regarded by Zuidema (1974) as a laborious programming task, was expressed as an Advice Table and a strategy checked out at a cost of only two man-days. More recently (Bratko, 1978) the whole mating procedure known from the chess books was compressed into a Table of only one rule, comprising 5 pieces of advice expressible as follows:

1. Look for a way to mate opponent's king in two moves.
2. If the above is not possible, then look for a way to further constrain the area on the chessboard to which the opponent's king is confined by our rook.
3. If the above is not possible, then look for a way to move our king closer to opponent's king.
4. If none of the above pieces of advice 1, 2, 3, works, then look for a way of maintaining the present achievements in the sense of 2 and 3 (i.e. make a waiting move).
5. If none of 1, 2, 3, or 4 is attainable then look for a way of obtaining a position in which our rook divides the two kings either vertically or horizontally.

The 'and-or' tree search, carried out by module 1 of the AL1 system when generating a forcing tree satisfying a corresponding piece of advice, is limited to a depth of 2 ply for pieces 2, 3 and 4, and to 3 ply for pieces 1 and 5. Quality of play was respectable by the standards of the chess books, never needing more than 25 moves to force the mate. The worst case minimax-optimal path length is known to be 16 moves (Clarke, 1977).

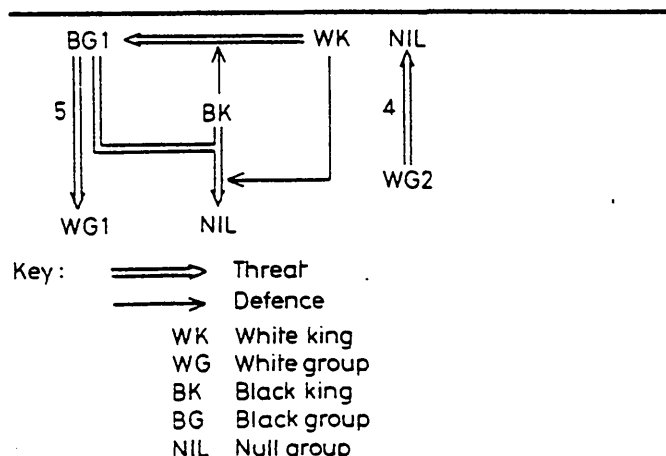


Fig. 3 The ADD corresponding to the position shown in Fig. 1. In the form described by Tan (1977) the ADD would not indicate the self-stalemate threat which the BK generates jointly with BG1. The above diagram is based on an extended notation which takes care of this. For a fuller account Tan's paper should be consulted. Integers denote the minimal number of moves required to carry out a threat

## HOW DIFFICULT IS THE KNKR PROBLEM?

From chess player's point of view:	}	<p>—Chess books: Keres, KNKR, 2 pages. Fine, KRKN, 8 pages. Longest variation in Fine before capture of the Knight: 24 moves; longest known variation 27 moves.</p> <p>—Tournament games: usually a comparatively easy draw, but there are examples where the weaker side went wrong (e.g. Neumann-Steinitz, 1870).</p>
From chess programmer's point of view:	}	<p>—Knowledge v. search:</p> <ol style="list-style-type: none"> <li>1. Rules of chess: <i>don't get mated!</i> Up to 24 moves before losing knight plus up to 16 moves before being mated:  <math>\approx 30</math> ply</li> <li>2. Additional advice: <i>don't lose knight!</i>  <math>\sim 48</math> ply</li> <li>3. Additionally: <i>keep king and knight together!</i>                      Necessary, and probably sufficient, lookahead is:                      10 ply</li> <li>4. <i>Advice contained in KNKR table.</i>                      To preserve draw, and conserve king-centrality:                      4 ply</li> </ol>

Fig. 4 Salient features of the KNKR end-game. The size of the total problem space, after reduction by disregarding symmetric cases, lies between  $3 \times 10^6$  and  $4 \times 10^6$ . The boxed figures show the depth of search necessary, for the program's given state of knowledge, to select a draw-preserving move. Ply = half move

### The KNKR game

In further experiments with the AL1 system, the king + knight v. king + rook ending (KNKR) was used as an experimental domain that is not trivial from the human expert's point of view. Fig. 4 presents some data that illustrate the difficulty of this end-game. The first two items, chess books and tournament games, give some insight into the difficulty from the *chessplayer's* point of view. The difficulty from the *programmer's* point of view is illustrated by the third item which indicates the relationship between the amount of knowledge possessed by the program and the depth of game-tree search required for correct play.

The KNKR ending is usually drawn, but under certain circumstances the stronger side (the one with the rook) can win. A winning procedure, when there is one, consists usually of combining three basic principles (Fine, 1964):

1. Create mating threats.
2. Force separation of king and knight.
3. Stalemate and capture the knight.

The KNKR advice table must cope with the above threats and thus preserve a draw for the weaker side when starting from a theoretically drawn position (in all such, the king and knight are not separated). The table, shown in Fig. 5, contains enough knowledge (according to experimental tests) both to preserve the draw in positions with king and knight sufficiently close together *and* to maintain the degree of centrality of the weaker side's king *while searching to a depth of at most 4 ply* (half-moves). Centralisation of the king is important to the weaker side because mating threats can occur only when the king is on the edge. Therefore when the king is on the edge the defence becomes considerably more difficult. The KNKR table thus actually conserves the degree of easiness of defence. When the king is started on the edge and not separated from the knight, for those positions which are theoretically drawn, the table preserves the draw in all cases tested. Recently a class of specially tricky positions has been discovered by D. Kopec, not previously known in chess literature, where the only correct defence requires a counter-intuitive separation

of king and knight. A correct treatment of such positions with king and knight separated would require additional knowledge.

The upper table of Fig. 5 specifies four rules (CR, R1, R2, ER) by the 'Yes, No, Don't-care' column patterns. These patterns refer to POP-2 predicates flanking the table:

- OKEDGE —our king on the edge;  
 OKONSEP —our king and our knight separated (distance greater than 4 in king moves);  
 CORNCASE—corner case, a special 'classical' situation (e.g. Fine, 1964) with the king in the corner, requiring exceptional treatment.

The current chess position is matched against the rule patterns from left to right. As soon as a match is found the corresponding rule is applied. For example, if the position does not satisfy the CORNCASE condition and the king is on the edge and the king and knight are not separated, then rule R2 matches the position and the list of pieces of advice 1, 5, 6, 7, 8, 12 is applied. The pieces of advice are defined by the lower table in Fig. 5. Consider for example Advice no. 1 called KILLROOK. The better-goal to be satisfied in them-to-move positions specified with this piece of advice is TRDEAD (their rook dead). As already stated, depth of search is limited to 4 ply, and to improve search efficiency we also specify holding-goals NOT ONLOST (not our knight lost without compensation) and TRDEAD OR CHECK (their rook dead or their king in check). By this the search is limited to consideration of immediate captures and checking-moves, which amounts to looking for ways of forking their king and rook. This tactic is indicated by the fact that there is no other way to force capture of the rook.

When testing the correctness of the table a variety of players, two of National Master strength (rated over 2300 on the international scale), have engaged the system in play for a total elapsed time of more than 10 hours (150 moves on each side, starting from different positions). No absolute way exists of proving correctness for all possible positions short of either (a) exhaustive checking through the total space of 3-4 million

		KNKR Table					rules								
condition predicates		OKEDGE	OKONSEP	CORNCASE	CR	RI	R2	ER							
					—	N	Y	—							
					—	N	N	—							
					Y	—	—	—							
					9	1	1	10							
						2	5	11							
						3	6	12							
						4	7								
						11	8								
							11								
							12								
		Better-goals					Holding-goals								
		us-to-move					them-to-move								
		OKONNDLT (utm)	TRDEAD (ttm)	NOT MATE	NOT ONLOST	ONSAFE2P	TRDEAD OR CHECK	NOT ONOKATT	OKEDNDGE	OKCONDGE	OKONDLE2	KINGSCLOSE	OKONDLE3	TKONDGTI	OKONDEQ4
pieces of advice	1: KILLROOK	—	Y	⊃	Y	—	Y	—	—	—	—	—	—	—	—
	2: HOLD1	—	—	⊃	Y	Y	—	Y	Y	—	Y	Y	⊃	—	—
	3: HOLD2	—	—	⊃	Y	Y	—	Y	—	—	Y	Y	⊃	—	—
	4: HOLD3	—	—	⊃	Y	Y	—	Y	—	—	Y	Y	⊃	—	—
	5: HOLDEDG1	—	—	Y	Y	Y	—	Y	—	Y	Y	—	⊃	—	—
	6: HOLDEDG2	—	—	Y	Y	Y	—	—	—	Y	—	—	⊃	—	—
	7: HOLDEDG3	—	—	Y	Y	Y	—	—	—	—	Y	—	⊃	—	—
	8: HOLDEDG4	—	—	Y	Y	Y	—	—	—	—	—	—	⊃	—	—
	9: CORNCASE	—	—	⊃	⊃	⊃	—	—	—	—	—	—	—	—	Y
	10: APPROKON	Y	—	Y	Y	Y	—	—	—	—	—	—	—	—	—
	11: SURVIVE1	—	—	Y	Y	Y	—	—	—	—	—	—	—	—	—
	12: SURVIVE2	—	—	Y	—	—	—	—	—	—	—	—	—	—	—

Fig. 5 The upper table is the KNKR Advice Table as written and tested. The integer lists index a repertoire of 12 pieces of advice: These are shown in the lower table expanded in the form of calls to indicated subsets of the 14 goal predicates listed along the top. 'Y's enclosed in brackets are logically implied by the other predicates selected in the same row. The symbols in parentheses (utm) and (ttm) mean 'us-to-move' and 'them-to-move' respectively

positions or (b) formal proof. AL1's tabular format offers simplifications which make the latter an attractive topic for study.

When playing the KNKR ending on the PDP-10 computer the present implementation of AL1 spends on average about one minute of computer time per move, mostly due to the comparative inefficiency of the forcing-tree generating routine. The program examines about 10 nodes in the game-tree per second. When run on comparable machines, other chess-playing programs, e.g. CHESS 4.5 (Slate and Atkin, 1977) or MASTER (Birmingham and Kent, 1977) examine at least a few hundred positions per second. A new version, AL2, is under construction with an eye to increased run-time efficiency, among other improvements. But considering AL1's efficiency from the point of view of programmer productivity, these experiments gave evidence of great savings. Table writing and check-out for KNKR occupied one of us (I.B.) for less than six weeks. We doubt whether correct play, especially if 'centrality preservation' is to be included, could be programmed using standard methods in less than a substantial multiple of this figure other than by promoting large proliferations of

forward search. As an annotation on this last remark, we append results, kindly supplied by D. Slate, of having the leading US tournament program CHESS 4.5 play the KNKR game against an expert opponent from selected starting positions.

**Performance of CHESS 4.5 tournament program with KNKR**  
Tournament programs have the aim of playing reasonably well, but not of course with guaranteed correctness, in all phases of the game: opening, mid-game, and ending. Such 'general' chess programs cannot pay much attention to specific features of different position-types. Rather these programs embody generalised chess principles, or heuristics, hopefully applicable to the large majority of possible positions. Lack of position-type specific knowledge is to some extent balanced by deep lookahead, facilitated by fast game-tree search routines, efficient tree-pruning, efficient coding, and fast hardware.

CHESS 4.5 was required to defend the weaker side of KNKR against a human opponent rated just over 2000 on the US Chess Federation scale, i.e. an 'expert'. The program ran on a CDC 6400 machine, on which it was able to win the 1976

ACM Computer Chess Championship (more recently it has had highly successful trials on the much faster Cyber 176). CHESS 4.5's general evaluation function was used without allowing any adjustment or special 'tuning' to the KNKR problem. Search depth was set to 7 ply. Since forced variations are searched beyond this pre-set horizon, moves 8 ply deep were occasionally searched in the present case. Under these conditions, CHESS 4.5 typically looked at a few tens of thousands of nodes per move and spent up to 120 seconds per move, typically between 30 and 60 seconds.

Three trials were made, using test positions taken from those used in the experimental validation of the KNKR table earlier described.

1. A 'classical' difficult defence (Fine, 1964; Keres, 1974) with the weaker side's king in the corner. CHESS 4.5 found correctly the move considered most difficult in the books, but then stumbled on the fourth move of the main 'book' variation, obtaining a lost position.
2. Another difficult position, with the weaker side's king on the edge (Keres, 1974). CHESS 4.5 found the only correct defence against the main line given by Keres (i.e. 8 best moves in a row).
3. A further position taken from our own tests, with the weaker side's king in the centre (easiest defence). CHESS 4.5 allowed its king to be driven to the edge resulting in a harder defence. This enabled the opponent to create mating threats, and after additional weaker moves by the program the king and knight got separated, leading to a lost position.

It is interesting to observe that the program's opponent, although an expert, after achieving theoretically won positions never grasped the opportunity actually to defeat the program. This has a bearing on the level of difficulty of this subdomain.

Conclusion: thanks to efficient tree-search, CHESS 4.5 was able to find a correct move in many difficult positions. But the

lack of specific advice: 'Keep king and knight together!' and 'Preserve the centrality of the king!' could not be entirely compensated by the efficient and comparatively deep search to 7 or 8 ply.

#### Discussion

It was pointed out by Shannon (1950) on general grounds, and more recently by Berliner (1974) on the basis of authoritative new theoretical and experimental work, that fast tree-search and uniform heuristics will not suffice for mechanising the highest levels of chess skill. In spite of impressive recent progress up the human tournament scale by 'brute force' programs the knowledge-gap from which these programs suffer still bars them from the higher reaches. The work here reported shows that the weaknesses inherent in the brute force style can be shown up even by quite a simple chess subdomain. The same subdomain, however, yielded readily to a more knowledge-oriented approach, for which the ALI system provided highly effective support.

The underlying formal model of the ALI problem-solver closely matches the basic structure of a range of combinatorial problems. Our experience supports the idea that the Advice Language methodology should be applicable to problems in such areas as algebraic manipulation, symbolic integration, robot plan-formation, and a variety of optimisation and scheduling tasks. While detailed accounts appear elsewhere (Bratko, 1978; Bratko and Michie, 1978), this brief overview was prepared in the hope of arousing interest among those actively engaged in one or another of such areas.

#### Acknowledgement

Thanks are due to the Research Community of Slovenia, to the British Council and to the University of Edinburgh for support and facilities.

#### References

- BERLINER, H. (1974). Chess as problem solving: the development of a tactics analyser, Ph.D. Thesis. Pittsburgh: Carnegie-Mellon University.
- BIRMINGHAM, J. A. and KENT, P. (1977). Tree-searching and tree-pruning techniques, *Advances in Computer Chess 1* (ed. M. R. B. Clarke), pp. 89-107, Edinburgh: Edinburgh University Press.
- BRAMER, M. A. (1977). Representation of knowledge for chess endgames: towards a self-improving system, Ph.D. Thesis, Milton Keynes: The Open University.
- BRATKO, I. (1978). Proving properties of strategies expressed in the ALI assertional language, (*Inf. Proc. Letters*, forthcoming.)
- BRATKO, I. MICHIE, D. et al. (1978). A representation for pattern-knowledge in chess end-games, *Advances in Computer Chess 2*, (ed. M. R. B. Clarke, forthcoming).
- CLARKE, M. R. B. (1977). A quantitative study of king and pawn against king, *Advances in Computer Chess 1* (ed. M. R. B. Clarke) pp. 108-118, Edinburgh: Edinburgh University Press.
- FINE, R. (1964). *Basic Chess Endings*, New York: David McKay Company.
- HUBERMAN, B. J. (1968). A program to play chess end games, *Technical report no. CS106*, Stanford University: Computer Science Department.
- KERES, P. (1974). *Practical Chess Endings*, London: Batsford Ltd.
- KMOCH, H. (1959). *Pawn Power*, New York: David McKay Company.
- KNUTH, D. (1976). Mathematics and computer science: coping with finiteness, *Technical report STAN-CS-76-541*, Stanford: Department of Computer Science.
- MCCARTHY, J. (1959). Programs with common sense, *Mechanisation of Thought Processes 1*, London: HMSO.
- MICHIE, D. (1976). An advice-taking system for computer chess, *Computer Bulletin*, Series 2, No. 10, pp. 12-14.
- SHANNON, C. E. (1950). Programming a computer for playing chess, *Phil. Mag.* (Lond.), 7th ser. 41, pp. 256-75.
- SLATE, D. J. and ATKIN, L. R. (1977). CHESS 4.5—the Northwestern University chess program, *Chess Skill in Man and Machine* (ed. P. Frey), pp. 82-118, New York: Springer Verlag.
- TAN, S. T. (1977). Describing pawn structures, *Advances in Computer Chess 1* (ed. M. R. B. Clarke), pp. 74-88, Edinburgh: Edinburgh University Press.
- ZUIDEMA, C. (1974). Chess, how to program the exceptions? *Afdeling Informatica, IW21/74*, Amsterdam: Mathematisch Centrum.

From Advances in Computer Chess 2, (1980)  
 ed. M.R.B. Clarke, Edinburgh:Edinburgh University Press  
 How Hard is the Play of  
 the King-Rook-King-Knight Ending?  
 D.Kopiec and T.Niblett

**ABSTRACT**

An exhaustive database of the KRKN ending revealed inadequacies in the published analyses. It was also used (see appendix 3) to demonstrate that the domain can be mastered by an endgame expert after special study. Experiments with Class 'A' players showed that the defence of the draw for the knight's side is in general easier than demonstration of the win for the rook's side, but that a special class of positions exist for which the correct defence runs counter to most players' intuition. The nature of these positions is examined, and modifications are proposed to existing KRKN theory.

**INTRODUCTION**

The subject of the first part of our paper is a database for the KRKN endgame. The second part reports experiments with human players in the same endgame. The third discusses the main concepts for the domain.

**USE OF A DATABASE TO INVESTIGATE THE KRKN DOMAIN**

By 'database' we mean a complete computer-stored look-up table for optimal play. Such a database has already proven invaluable for the study of KPK (Clarke 1977). With its help Bramet (1977) has proved the optimality of a program to play this endgame. Using a database for KRK, Clarke (1977) has shown that White can force a win from any position in 16 moves or less - instead of 17 as was previously thought (Fine 1941). These endings however are ones for which Chess Masters have no difficulty in finding correct, if not optimal moves. We think this is due to the small number of 'patterns' needed to assess any position statically, despite the longest win being 19 moves for KPK, 16 for KRK. Very little lookahead seems to be needed for these endings.

KRKN is different: here we enter a domain that can be challenging even for a master. This is a subgame of chess which is possible to compute fully by machine (a database), but very hard for the unaided person to play correctly. The ending KQKR illustrated a similar case recently when two International Masters repeatedly failed to find a winning line in play against a data base for this ending. We have had similar results with KRKN against master-strength players.

Other databases already constructed are those for KRPKR and KQPKQ (Arizarov and Futer 1979), the latter being notoriously difficult even for leading Grandmasters. Last year Grandmaster Bronstein, once challenger for the world championship, had a position with KQPKQ at

adjournment and used the database to find a win he himself could not. Although KRKN is simpler than this, one of our later examples shows the breakdown of 'pattern matching' even for a Grandmaster. The major difference then between this endgame and KPK is the need for very much more tree-search, or lookahead by humans.

KRKN databases have a fairly long history. The first of which we are aware was completed in 1970 by Thomas Strohlein (1970) as part of a Ph.D. Dissertation on Graph Theory and Combinatorics. Recently Ken Thompson of Bell Labs computed databases for all the interesting four-piece endings including KRKN, and Gams (1978) has computed specifically the KRKN case and has partially validated Thompson's work. Finally at the M.I.R.U. we have independently constructed one. With this we can check one database against another to ensure correctness. This is a major problem with databases: because of their size only machine checking is possible and even this is daunting. In the case of Grandmaster Bronstein's consultation of the KQPKQ database, the actual line he was given would have resulted in a draw due to a database error at the very end!

Algorithm for Database Construction

Michael Clarke (1977) has already described a procedure for KPK. However, KRKN differs slightly, for breadth-first backing up is used. The procedure can be conveniently divided into three parts (see figure 1).

All positions for White where he can mate immediately or win the black knight in one move are defined as won at depth 1. The first step is to find all these without loss of the rook or stalemate. For each legal BTM position a counter is set for the number of illegal successors. This completes the initialisation.

Backing up from WTM positions won at depth N is done by finding all legal predecessors of the position. Since there are  $8 + 8 = 16$  potential moves in all, the counter will show the value 15 at the moment that the set of legal moves is exhausted. If any hasn't previously been marked as won, then if the counter has value 15, that position is marked lost depth  $N + 1$ ; otherwise the counter is incremented by 1.

Backing up from BTM positions is simpler. Again all legal predecessors are found. Any not previously marked as won are marked as won depth N.

Results

The procedure described above was executed as a program on the DEC System-10 computer. The resulting database was used to partially check the correctness and completeness of Thompson's KRKN database. The latter was the one actually used for most of the results now to be reported.

It is difficult to estimate the exact number of legal positions. Due to symmetry it is much less than  $64^4$ . Thompson (private communication) estimates 741 000 drawn positions and 651 000 positions won for White in the canonical state space.

Note that the backing-up method described entails that only won

A White-to-move (WTM) position is won depth = 1 if White can mate or win the knight without stalemate or loss of the rook

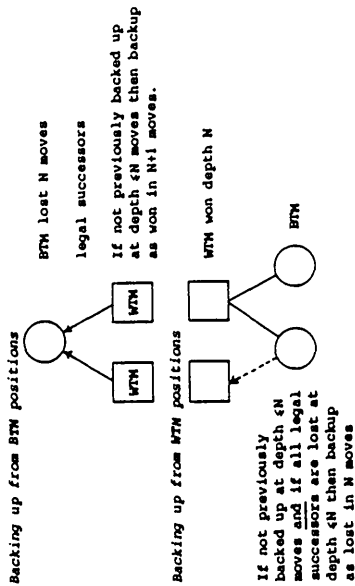


Figure 1

Depth of loss (moves)	No. of positions
1	378,518
2	95,450
3	46,269
4	30,729
5	20,055
6	15,071
7	11,740
8	9,495
9	8,582
10	7,415
11	6,308
12	5,356
13	4,133
14	3,356
15	2,290
16	1,621
17	1,333
18	1,046
19	727
20	556
21	458
22	373
23	302
24	178
25	111
26	18
27	2

Figure 2. The distribution of losses with depth

positions are assigned a depth in the database. Draws are recognised by lack of this marker. There is, incidentally, one position lost for White! There are therefore roughly 1 400 000 legal positions with White to move. This is about 15 times larger than KPK. The distribution of losses with depth, WTM, is shown in figure 2.

The two extreme positions lost in 27 moves with optimal play are:

WK:d1 WR:h1 and WK:c1 WR:f8  
BK:b1 BN:g4 and BK:a3 BN:e2

(For optimal winning sequence see appendix 1.) It is important to realise that these winning lines are far longer than needed for any position given in the literature. The deepest win we have been able to find as a published position is 16 moves with optimal play. The database can be used to generate (a) optimal play for White from any won starting position, (b) optimal play for Black from any lost starting position, (c) correct play for Black from any drawn position. Safe knight-capture is counted as 'win' without considering play of the residual KRK game. This section concludes with a discussion of KRKN as it is represented in the literature.

#### Analysis of Previously Published Treatments of KRKN

The history of KRKN is peculiarly long, peculiar in that the king, rook and knight move in the same way as in the ancient game of Chaturanga, which is otherwise very different from Chess. Several theoretical positions can be dated to the ninth century AD. In English the two most comprehensive treatments of the endgame are in Fine (1941) and Averbakh (1978). Fine's analysis is based on that of Berger whereas Averbakh relies on many more sources. Fine however gives a more detailed analysis for each position.

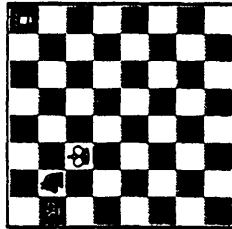


Figure 3. Black to move

To show the difficulty of the ending and its 'opacity' even in the face of extensive analysis we discuss the position shown in figure 3. This position is a variant of that given for Chaturanga by al-Adli in the ninth century. It was rediscovered in 1859 and (incorrectly) analysed in 'The Chess-Players Chronicle'. Berger subjected the position to detailed analysis, and this was used by Fine in 'Basic Chess Endings' and continued by several analysts. The database shows that it can be won in 14 moves.

To show the difficulty of analysis we give Fine's main line (his

#### How Hard is the Play of the KRKN Ending?

punctuation) for the first few moves, comparing his move with that of the KRKN database:

1	...	Nb5+	forced
2	Rb5	Nb7	all optimal
3	Rb1	...	Rb5 is best, one move shorter.
		...	How-
		...	ever Kc6 wins, and more quickly than Fine's main line. This is one of two
		...	cases where Fine gives the wrong game-theoretic value for a position.
3	...	Nd6+	forced
4	Kc6	Nc4	
5	Rd8!	...	In spite of Fine's exclamation

mark, this move changes the depth of win from 11 to 17 moves in this position.

This demonstrates a typical phenomenon. The human master finds a plan which preserves the game-theoretic value of the position, but which is inefficient in the minimax sense.

We have analysed all the main lines given in Fine, and figure 4 illustrates the results. It is superficially surprising that there are so few non-optimal moves at large depths and so many at relatively shallow depths. However, if the win is very deep there is usually only one good winning move. At depths of 7-10 moves the master doesn't try in his tree searching to find optimal moves, just good ones. At lower depth still he can see through to the end.

Using the database we have seen that optimal play is very hard to find, even for the analyst. There is a small subset of the position space, the longer wins (17 to 27 moves deep), which has never been explored. Masters have great difficulty in winning these positions. In our small-scale trials of human play against Thompson's database (not with full tournament time allowance) we never saw it done. Subsequently, however, similar but more systematic tests were convincingly passed by A.J. Roycroft, not a Master but a specialist in the chess endgame study. His own account is reproduced as appendix 3.

#### EXPERIMENTS WITH CLASS 'A' PLAYERS

Further information about the nature of the difficulties was obtained from experiments with Class 'A' players described below, in which the database did not figure.

##### Experiment 1: Weak Class 'A' Players

*Objective:* This experiment was to discover:

- How hard is it for an 'A' player to win a won position?
  - How hard is it for an 'A' player to draw a drawn position?
- Usually when humans are asked about this ending they will say either that they know nothing about it, or that it is drawn by keeping the king and knight close together and avoiding the edge of the board where mating threats can occur. Since Class 'A' players are better than average tournament

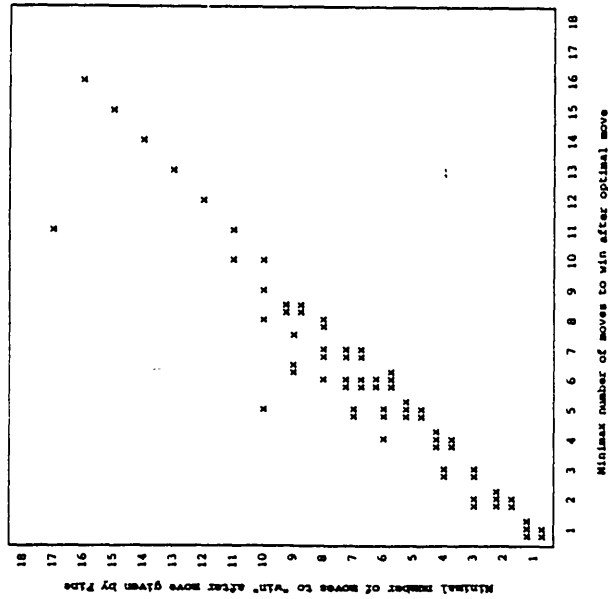


Figure 4a. All positions White to move

players, and are likely to know just the two above concepts, their performance on both the stronger and weaker side, under tournament-like conditions, was thought worth investigating.

*Experimental Design.* Three subjects, all university students averaging 1882 in ELO rating, were each assigned two positions, one drawn, which they were to defend with the knight's side, and the other won which they were to win for the rook's side. Drawn positions included starting positions from which testing was done on Ivan Braiko's Advice Table (see Braiko and Michie, this volume). All these positions have in common the mutual proximity of the defending K and N. However in 'Corncase', (see figure 5), after 1 Rb2+ Ka1, 2. Rb8, we reach the only position which Braiko had to program as a special case, since only 2 ... Ne2, separating K and N as much as possible in the position, draws.

Won positions were selected as the longest (not necessarily main-line) variations given in Fine's *Basic Chess Endings*, reportedly requiring 24, 17 and 15 moves to win (see figure 5, T4, T5, T6). Later checking of these positions with the Thompson database revealed that they actually require 14, 9 and 11 moves to win, with optimal play for both sides. Subjects were

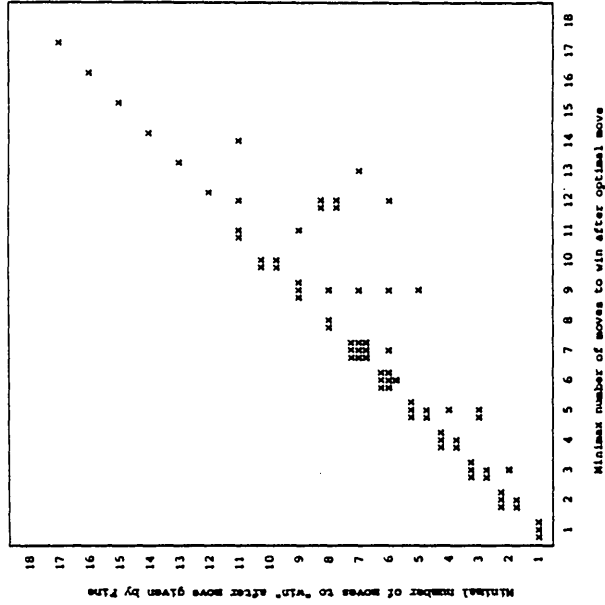


Figure 4b. All positions Black to move

assigned positions based on a predetermined rank-ordering of their difficulty. Proficiency was demonstrated by holding the game-theoretical value over 30 moves of play in one hour against a U.S. Master opponent (D.K.). Otherwise resignation, imminent checkmate, or a loss of material were the termination conditions.

*Results and Conclusions.* These Class 'A' human subjects had no difficulty defending typical drawn KNKR positions except for 'Corncase', where the subject later testified that he had played the correct 'separating' move without knowing why the other moves lost. The experimental design and results are summarized by figure 6. On the stronger side subjects failed in two of three cases to win. The one subject who did win, did so in 11 moves, 6 less than in the variation given by Fine, but Fine gives 17 moves for T5, i.e. 6 more than 11! A later check against the Thompson database showed that optimal play requires 9 moves (see appendix 2).

The database also showed that the experimenter's knight's-side defence in the experiment was suboptimal in the minimax sense.

Immediately following the experiment, a complete analysis was done to discover all game-theoretic value-changing moves (i.e. 'mistakes'). Since



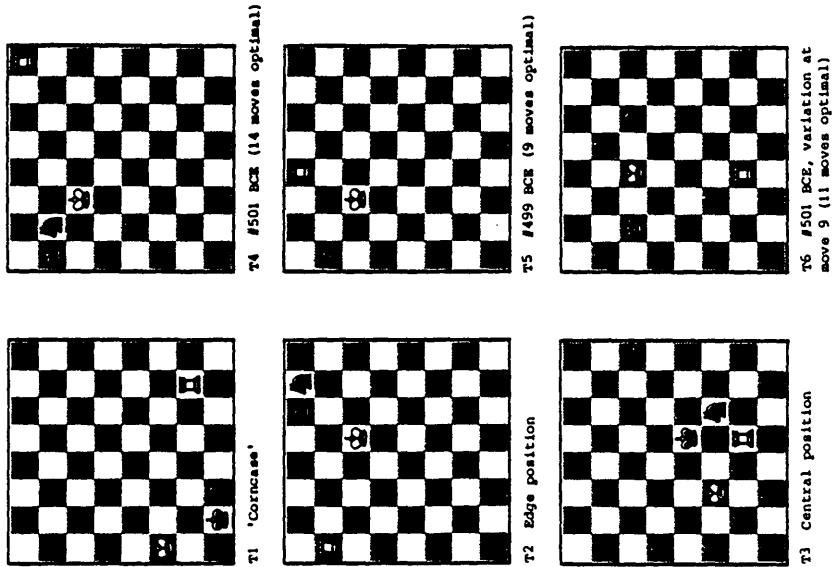


Figure 5. The positions for Experiment I. T1, T2, T3 are drawn positions and T4, T5, T6 are won positions. (BCE stands for Basic Chess Endings by R. Fine.)

subjects held the draw in all three cases, it was only a question of whether the human opponent had failed to exploit some error which they had made, but no errors were found. When subjects failed to win won positions, their errors occurred quite early in the sequences. The retrospective analysis of the continuation from position T6 proved to be most interesting. After: 1 Rb2+ Ka5 2 Kc6? (2 Rc2 was correct) Ne4, is this position a draw or a win for White?

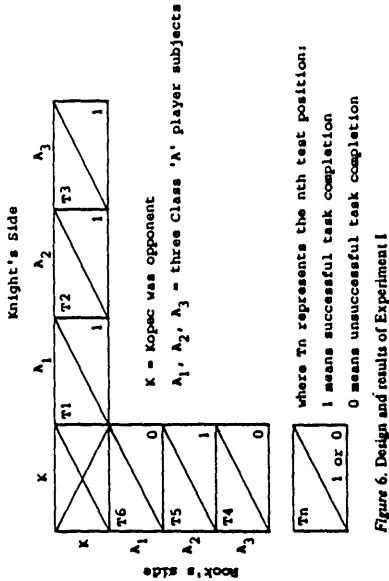


Figure 6. Design and results of Experiment I

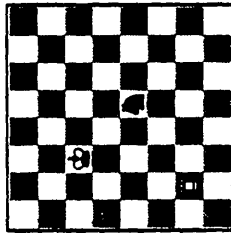


Figure 7. Continuation from T6 of Experiment I with subject P. Edwards. Is this position a draw or a win for White?

At first 1 Rb3! was tried, when not 1 ... Ka4? 2 Rd3! and Black is lost, but 1 ... Nd2 2 Rd3 Nb1! etc. when the position is clearly drawn. Then 1 Kd5 was tried with some very surprising variations to follow, but we will let readers study the position longer before giving the variations which provide the answer. This retrospective analysis led to the discovery of three positions where counter-intuitive 'separating' moves were required to draw and the Bratko Advice Table erred. The failure of the Table for these positions was not really surprising, for it had been designed for positions where the BK and BN were relatively close together (i.e. < 3 king moves apart) with the belief that all positions where K and N are further separated are either lost or, if not, K and N must approach each other immediately. This is the view which had been presented in all the basic chess literature for this ending. Positions where K and N are reasonably separated were considered lost. Only recently, thanks to Mr A.J. Roycroft, have we learned of a Czech paper by Mandler (1970), which indicates that work has been done on this aspect. These positions, which at the time were

believed to be new to the theory of this ending, led to an entire benchmark of 16 positions where K and N are separated. Some of these positions will be discussed in the last part of this paper, but now we will discuss their importance in Experiment II.

#### Experiment II: strong Class 'A' players

**Objectives.** Subjects' performance in the defence of KNKR in Experiment I pointed to the conclusion that it was not a difficult task for Class 'A' players. However the retrospective analysis brought new interest to the problems of correct play in drawn positions with the domain much richer than previously thought. Experiment II was seen as necessary to find out just how hard is the defence of KNKR when the domain is enriched with these 'new' positions, or more specifically: How well will strong Class 'A' human players defend KNKR positions where counter-intuitive 'separating moves' are required as well as other concepts?

**Experimental Design.** The 16 positions generated by the retrospective analysis of Experiment I and tests on the Braiko Table included 13 positions which the table did not handle correctly. For Experiment II nine of the hardest KNKR positions were selected, with highest ranking in difficulty going to positions where counter-intuitive 'separating' moves were required. Further ranking was based on the number and complexity of concepts considered necessary for correct play. Three Class 'A' players, with ratings of 1950, 1975 and 1915 were each allotted three positions in such a way as to approximately even out the mean ranking of difficulty and to include for each subject a reasonable diversity of concepts necessary for correct play (figure 9). The nine positions used are given in figure 8. Viewing them from left to right, top to bottom, we go from the position highest ranked in difficulty (D3) to the lowest ranked (D15). Note that D3, D2, and 'Corncase' all require counter-intuitive 'separating' moves, and thus are highest ranked. Since D3 includes D2 as a subset of correct play from it, but first requires a 'natural' knight move, it is higher ranked than D2. The method by which these nine positions were ranked is given in figure 9. Of these nine positions the Braiko Advice Table had failed to play the correct move in seven of them, handling 'Corncase' and D15 correctly.

Subjects and experimenter (D.K.) were allowed 40 minutes in which subjects were to play 20 moves which held the draw. The time had been slightly curtailed compared with Experiment I since once subjects succeeded in playing the first few moves correctly the task became easy and rather pointless. According to their post-mortem testimonies, two of the three subjects in this experiment knew that they should keep K and N together.

**Results and Conclusions.** As shown in figure 10, subjects failed in 6 of 9 trials. Failure in 4 of the 5 positions highest ranked in difficulty indicates that our *a priori* notion of what comprises a difficult KNKR position for the human was probably correct. However subjects played the first unforced move correctly in 5 of the 9 positions. In addition, we note that despite subjects' correct handling of 'Corncase' in both Experiment I and

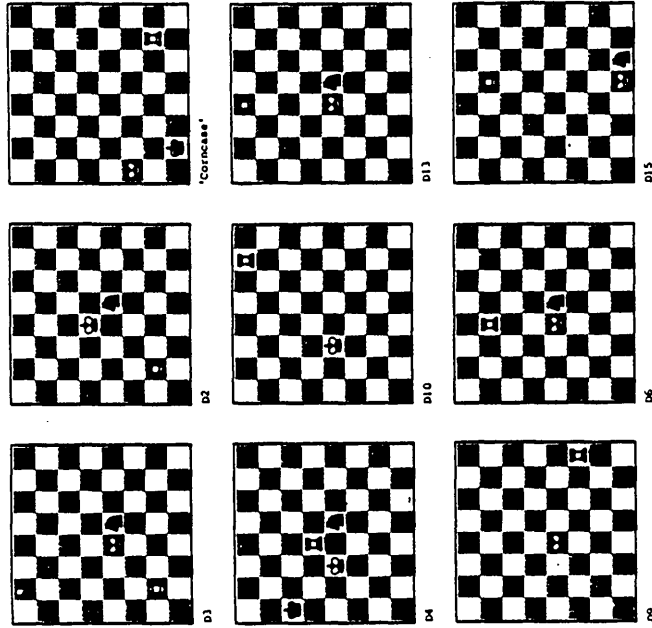


Figure 8. The nine positions used for experiment II, with D3 ranked hardest, going left to right, top to bottom to D15 ranked lowest. (Black to play in each except 'Corncase'.)

Experiment II, it is apparent from post-experimental questioning that subjects did not do the necessary calculation to see why the 'separating' move, 2... Ne2 was the only move which did not lose.

This experiment substantiated our notion that at least expert and perhaps master strength human play may be required for the correct defence of KNKR. We have also gained insight into specific concepts which may be necessary for correct play in positions where defending K and N are separated.

Positions were allotted to subjects in such a way as approximately to even out the mean ranking of difficulty, and at the same time to include for each subject a reasonable scatter. Symbols such as OKON, ONLOST2P, etc., are explained in the glossary below:

APPROKON: 'approach our king or knight'. Piece of advice needed to guide our knight or our king in separated positions

Position name	Rank	Concepts Involved	Subject treatments	Average rank
D3	1	APPROXON 7		
D2	2	7	A <sub>1</sub> :D3,D6,D10	4.67
Corncase	3	OKONDE4 (special case)		
D4	3	APPROXON COMMON GROUND		
D10	5	ONLOST2P OKONNLE	A <sub>2</sub> :D2,D13,D15	5.33
D13	6	OKTRATT ONLOST2P		
D9	7	APPROXON ONTRNDLE †		
D6	8	OKTRATT	A <sub>3</sub> :D4,Corncase,D9	4.33
D15	8	OKTRATT		

\* These are positions which require "separating" moves whose underlying concepts we have been unable to specify.  
 † Note that during Experiment II it was discovered that if White plays Kc4 after 1 ... Nc6† the addition of OKONNLE would be necessary.

Figure 9. Ranking of positions for Experiment II (average rank 4.77)

ELO RATING			
D3	D6	D10	1950
0	0	0	A1
X			
D2	D13	D15	1975
0	1	1	A2
X			
D4	CORNCASE	D9	1915
0	1	0	A3
X		X	

D1 = a drawn benchmark position  
 1 = task successfully completed  
 0 = task unsuccessfully completed (subject lost or resigned)  
 X = first unforced move was a losing one

Figure 10. Design and results of Experiment II. Kopec was experimenter in each case.

OKONDE4: 'our king our knight distance equals 4'  
 Common Ground: Potential meeting squares for K and N in separated positions.

ONLOST2P: 'our knight lost 2 ply'. Predicate which detects nearly all losses of our knight in two ply. Applied as 'Not ONLOST2P'.

OKONNLE: 'our king our knight new distance less than or equal to'. Means that we are happy to maintain or decrease the distance (in king moves) between our king and our knight, but not increase it

ONTRNDLE: 'our knight their rook new distance less than or equal to'. Intended to prevent our knight from approaching their rook in separated positions

OKTRATT: 'attack their rook two ply'. Predicate for positions where our king threatens to capture their rook thereby allowing common ground or some other concept to be attained

CONCEPTS INVOLVED IN

A BENCHMARK OF 20 DRAWN POSITIONS

Counter-Intuitive 'Separation' and Common Ground

In this section we discuss some of the positions used in Experiment I and Experiment II, as well as a few others.

On the basis of the insight gained we augmented the set of 16 positions mentioned earlier with a further four with the aim of establishing a benchmark of 20 positions. These should specifically test all the features of difficulty of which we were aware. Returning to figure 7 from the experimental run with subject P. Edwards, after 1 Kd5 we reach the position labelled D2 (see figure 8). To answer our earlier question, 'What is the value of the position in figure 7?' we now need a 12-ply search, i.e. 1 ... Nc3+, 2 Kc4 Ne4, 3 Re2 Nd6+ (any other N move gets the knight stranded as well), 4 Kc5 Nb7+, 5 Kc6 Nd8+ (if 5 ... Ka6, 6 Ra2+ Na5+, 7 Kc5 and wins), 6 Kc7 Nf7, and now the knight is clearly stranded. Instead, in position D2 the only drawing move is 1 ... Nf6!! We shall give Black's responses to all of White's possible K moves from figure 11, in counter-clockwise order: if (a) 2 Kc6 Ne4 (b) 2 Kd6 Ne4+ (c) 2 Kc6 Ne4 returning to figure 7 (d) 2 Kc5 Nd7+ or Ne4+ (e) 2 Kc4 Nd7 (f) 2 Kd4 Nd7 (g) 2 Kc5 Nd7+. Re-turning to D2 (figure 8), after 1 ... Nc3+, 2 Kc4 Ne4, if White plays 3 Kd4? instead of 3 Re2, we have D3 (figure 8), when only 3 ... Nd6 draws. Then if 4 Kc5 Ne4+, 5 Kd5, we return to D2, and so D3 is a superset of D2, whereby first the 'natural' Nd6+ is required, and later the 'counter-intuitive' Nf6+!! (D2). Considering D2, after 1 ... Nc3+, 2 Kc4 Ne4, we try 3 Rb5+? Ka6, 4 Rd5 reaching D4. Normally White's last few moves comprise part of a winning technique which entails: (1) separate BK and BN, (2) drive N further away, (3) trap N and win it. However in D4, Black has the drawing move Kb7 threatening Nf6 ... Ne8, ... Nc7 or ... Nf6, ... Kc7, and Nd7. (Other threats are ... Kc6 or ... Kc7 and ... Nd6, or Nf6 and Nd7 to follow.) White's inability to prevent all these 'paths' by which BK and BN threaten to meet, ensures that Black can draw. We have labelled these meeting squares 'Common Ground'. Figure 12 depicts this concept for D4

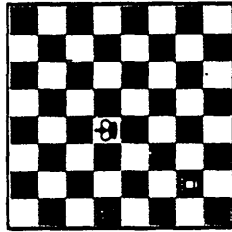


Figure 11. Position arising from D2 after 1 ... Nf6+!! This is the first case from our experimental work where a counter-intuitive 'separating' is required.

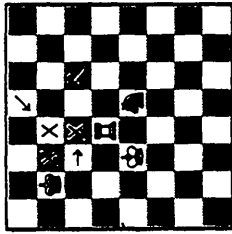


Figure 12. Position after 1 ... Kb7 from D4. Squares which are 'Common Ground' or potential meeting squares for BK and BN are marked with an X. Paths to these squares are marked with arrows also indicating approximate direction to target square(s).

after 1 ... Kb7, 'paths' denoted by '+' and the Common Ground denoted by 'X'. Another heuristic indicating the move ... Kb7 is 'if K and N are 3 or more K moves apart, don't move your K onto a file or rank where N can move'. Otherwise that N move can be met by a R move forking K and N.

If in D4 Black tries 1 ... Kb6?, then 2 Rd4! wins, i.e. 2 ... Nc5 and 2 ... Nf6 both lose to 3 Rd6+ and after 2 ... Ng5, 3 Rg4 Nf7, 4 Rg6+ Ka5, 5 Kd5 Nd8, 6 Rg8 Nb7, 7 Kc6 and wins.

In 'Corncase', the refutation to Black's other N moves after 1 Rb2+ Ka1, 2 Rb7 deserves more attention than the continuation after the correct move 2 ... Ne2. This might go: 2 ... Na2, 3 Kb3 Kb1 (if 3 ... Nc1+, 4 Kc2 wins), 4 Rb8!!.

In post-mortem discussion those two subjects who played the correct 2 ... Ne2 did not mention the passing move 4 Rb8 as the refutation of Na2. The idea of a 'passing' move is the only way White can win in the position. Then on 4 ... Nc1+, Kc3+ with 6 Kc2 to follow, wins. 2 ... Nd3, 3 Kd3 and then if 3 ... Nc5+, 3 ... Nb2, or 3 ... Nc1+, 4 Kc2 wins quickly.

#### 'Guided Knight Tours'

Next we shall consider D9 and D10 (figure 8). In D10 Na4 and Ne4 are the candidate moves. Ne4 is closer to the BK than Ne4 but on the edge, while the latter move is more centralized. In fact after 1 ... Na4, 2 Kb4 forces 2 ... Nb6 since Nc3 loses to 3 Rg2+ (ONLOSTP), and then Black's N is stranded with further separation from the BK to follow. Correct is 1 ... Ne4 when White can make no progress, though Black must still play very carefully, i.e. if 2 Kd3 Nc5+, 3 Kc4 repeats the position; or if 2 Kd4 Nd2 (not Nc3? Rg2+, etc.) 3 Rg2 Kc1 and we have the typical 'edge draw'. In D9 the candidate moves are obviously Nf5+ and Nc6+. Both moves bring the N a distance of five king moves from the BK; however Nf5+ offers

#### How Hard is the Play of the KRKN Ending?

no further path back to the BK. After 1 Nf5+, 2 Kc5 Ne7, 3 Rh6 the N is stranded and soon will be lost. Even after the correct move, 1 ... Nc6+, 2 Kc4, the BN still needs careful guidance — is 2 ... Na5+ or Ne5+ correct? This decision and the one on Black's first move can be guided by the important underlying heuristic 'When N and K are separated, don't move the N towards both the opposing K and R unless it is the only way back to your K'. Thus here the correct move is 2 ... Na5+ and after 3 Kb4 Nc6+ it is clear Black cannot be forced to further separate K and N. If 2 ... Ne5+?, 3 Kd5 Nd7, 4 Rh6 and again the N is stranded.

#### OKTRATT for Common Ground

The last group of positions from Experiment II to be discussed is D13, D6 and D15 (figure 8). These all share the idea of OKTRATT (our king their rook attacks) in order to allow 'Common Ground' to be achieved.

In D13 there are two drawing moves, 1 ... Kc7 and 1 ... Nd2. 1 ... Kc7 is clearer joining K and N next move. If 1 ... Nd2 then 2 Rb8+ Kc6! draws. Similarly, in D6 1 ... Kc8 allows 'Common Ground' to be achieved via either ... Nf6 or ... Nd6. This is the only move since 1 ... Nf6? is met by Rd6 forcing further separation. In D15 Black's 'normal' move, 1 ... Ng3 is thwarted by 2 Re3 when 2 ... Nf5 is impossible due to the fork, 3 Re5+ and thus the knight is quickly lost. In order to avoid the fork and still allow a path for 'Common Ground' via 2 ... Ng3 or 2 ... Ne3, depending on how White plays, the 'separation' move 1 ... Kc6 draws. The more obscure 1 ... Nh2 also draws because White cannot cut off both the paths ... Nf3 and ... Ng4 to follow.

D8 (figure 13) was contrived to show that the distance between K and N is not as important as the availability of paths between them, especially for the N. Both the moves ... Nb5 and ... Nc6 bring the BN closer to the BK, though at the same time closer to the WR and WK. Yet these are the only moves to consider, for a K move would be too slow in joining BK and BN, i.e. 1 ... Ke2, 2 Kc4! Nc6, 3 Rc8+ and the N is soon trapped. Comparing 1 ... Nc6 and 1 ... Nb5, we see that the former is more centralized, offering two clear paths through the centre (... Ne5 and Nd4) back to the BK. White cannot prevent both these 'threats' and thus Black draws easily with 1 ... Nc6. On the other hand, 1 ... Nb5 only offers ... Nd4+ as a direct path back to the BK, with ... Nd6 as an indirect route. The natural move, 2 Rd8, cuts off these two possibilities (see figure 13, D8.2). Yet ... Nc7 draws! After 1 ... Nb5! from D8 it was later found that 2 Kc4! Nd6+, 3 Kd3 wins. From D8.2 White is threatening Rd7 and then Kb4 winning the N. Black's only way out of this is to play ... Nc7 immediately, before the N gets 'fenced in'. The N wanders, persistently evading the R. If the WK tries to participate, the BK comes to the rescue of the N just in time, or the N escapes back to the BK. From D8.2 if we continue 1 ... Nc7, 2 Rd6 Kf2, 3 Rf6+! Kg2 (the only move — if 3 ... Ke3, 4 Kc4 and 5 Rc6 wins quickly) 4 Kc4, then the position D18 is reached. Here only 1 ... Kh4 draws. The explanation for why 1 ... Kg4 loses can be given as an extension of the heuristic given for D4, 'if K and N are separated by a distance of 3

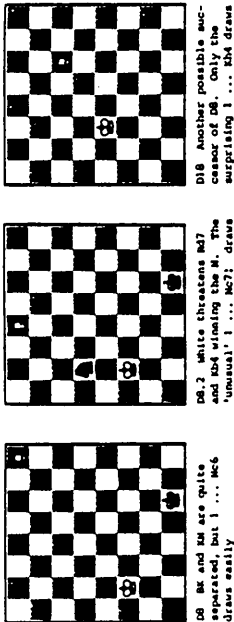


Figure 13. Some other positions from our benchmark. DB.2 and D18 are successors of DB.

or more K moves, don't move your K onto a file or rank which the N may need to occupy in two moves', i.e. 1...Kg4?, 2 Rc6 Ne8, 3 Re6 Ng7 (3...Nc7, 4 Re7 wins the N soon), 4 Rg6+ and wins.

#### CONCLUSIONS

How hard is the KRKN ending?

1. The use of an exhaustive database has shown that published treatments of KRKN are marred by serious inaccuracy and incompleteness. Hence the domain cannot be described as easy, even for the chess analyst.
2. The attacking side's task in won positions is too hard for successful play to be reliably demonstrated by Masters against optimal defence. But a celebrated endgame study specialist was able to do this after intensive preparation on a benchmark of six 24-movers, only three of which had been made available to him for prior study (see appendix 3).
3. Play of the defence of king and knight against king and rook in drawn positions is easier than play of the attacking side in won positions. But even the defence proved to be too hard for strong Class 'A' players to conduct reliably.
4. Most of the difficulty of the defender's task is concentrated in a relatively small class of positions which demand counter-intuitive moves separating king and knight.
5. The key concept of separation between king and knight should not be measured geometrically as has been customary. A revised definition should be based on multiplicity as well as length of a available paths between the two pieces.
6. The KRKN ending is a great deal harder than has been assumed, and its complete codification has eluded chess theorists. Improvement of existing theory is here put forward. This was substantially aided by availability of the KRKN database.

#### ACKNOWLEDGEMENT

This work was done by the authors as graduate students in the Machine Intelligence Research Unit at Edinburgh University under the

supervision of Professor Donald Michie, whose substantial assistance in the preparation of this paper is hereby acknowledged. One of us (T.N.) also acknowledges financial support from a Science Research Council studentship. Dr Raymond Carhart gave valuable programming advice during the construction of the database.

#### REFERENCES

- Arlazarov, V.L. & A.V. Futer (1979) Computer analysis of a Rook endgame, in *Machine Intelligence 9* (eds. J.E.Hayes, L.I.Mikulich & D.Michie). Chichester: Ellis Horwood, and New York: John Wiley.
- Averbakh, Y. (1978) *Rook against Minor Piece Endings*. Batsford.
- Bramer, M.A. (1977) Representation of Knowledge for Chess Endgames: towards a Self-Improving System. *Ph.D. Thesis*. Millon Keynes: Open University. See also this volume.
- Clarke, M.R.B. (1977) A quantitative study of king and pawn against king, in *Advances in Computer Chess 1* (ed M.R.B.Clarke) pp.108-18. Edinburgh: University Press.
- Fine, R. (1941) *Basic Chess Endings*. New York: David McKay Company.
- Gams, M. (1978) Constructing complete game strategies. *Undergraduate thesis*. Ljubljana: University of Ljubljana Faculty of Electrical Engineering (in Slovenian).
- Mandler, A. (1970) *Studie. II.Svazek*. Prague: Edice Sachoveho Klubu Ustredniho Domu Armady.
- Strohlein, T. (1970) Untersuchungen über Kombinatorische Spielen. *Ph.D. Thesis*. Munich: Technische Hochschule München.

Appendix 2

When given the position WK:c6, WR:d8, BK:a7, BN:c3(#499, BCE) (see figure 5, T5) for the stronger side in Experiment 1, the subject, Pat Coleman (1965 rating), improved on the variation given by Fine by 6 moves. What follows is the experimental record:

W: P. Coleman B: D. Kopec	Minimax-Optimal Value (Moves)	Optimal move(s) and their minimax-optimal path-lengths
1. Rd4 (13)*	9	
1. ... Nf5	6	Kb8(9)
2. Ra4+(15)	6	
2. ... Kb8	5	
3. Re4	5	
3. ... Ka7	3	Ng3(5)
4. Kd5	8	Kc7(3)
4. ... Kb6	8	Nb3, Kb7, Kb6
5. Ke6(21)	7	
5. ... Ng3	7	
6. Re1	6	
6. ... Kc5	5	Kc7, Kc6(6)
7. Ke5 (22)	4	
7. ... Kc4	4	
8. Kf4	3	
8. ... Nh5+	3	
9. Kg5	2	
9. ... Ng3	2	
10. Kg4 (25) and captures N		

\* Figures in brackets after White's moves in this column are the total time from the allotted hour, consumed by subject.

Appendix 1. Optimal Move Sequences for the Two Longest Wins

Position: WK:d1, WR:h1, BK:b1, BN:g4  
 1 Rh4 Ne5 2 Re4 Nf7 3 Rb4+Ka2  
 4 Kc2 Ka3 5 Kc3 Nd6 6 Rb6 Ne4+  
 7 Kd3 Nf2 8 Kc4 Nd1 9 Rb3+Ka4  
 10 Rf3 Nb2+ 11 Kc3 Ka3 12 Rg3 Ne4+  
 13 Kc4 Ka2 14 Kb4 Nb2 15 Rg4 Nd3+  
 16 Kc3 Nc5 17 Rc4 Ne6 18 Ra4+Kb1  
 19 Ra5 Ng7 20 Re5 Ka2 21 Kd4 Kb3  
 22 Kd5 Kc3 23 Kc6 Kd4 24 Kd6 Kd3  
 25 Ke7 Kd4 26 Rg5 etc.

Position: WK:c1, WR:f8, BK:a3, BN:e2  
 1 Kd2 Nd4 2 Kc3 Nb5+ 3 Kc4 Nd6  
 4 Kc5 Nb7 5 Kb6 Nd6 6 Rf4 Kb3  
 7 Kc5 Nb7+ 8 Kc6 Nd8+ 9 Kb5 Ne6  
 10 Rf3+Kc2 11 Kc4 Kd2 12 Rf5 Kc2  
 13 Rf2+Kd1 14 Kd3+Nc5+ 15 Kd4 Nb3+  
 16 Kc3 Ke1 17 Rb2 Nc5 18 Kd4 Ne6  
 19 Ke3 Kf1 20 Rb6 Nc7 21 Ke4 Kf2  
 22 Ke5 Kc3 23 Rb7 Nd6 24 Kd6 Kd4  
 25 Rb6 Ne5 26 Rb4 etc.

Many positions, for either side, have more than one optimal continuation. The above two lines should therefore be regarded as specimen paths excerpted arbitrarily from two optimal-strategy trees.

Appendix 3

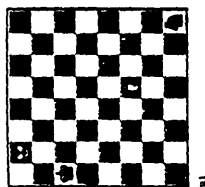
The following notes are contributed by A.J. Roycroft, author of *Test-tube Chess* and editor of *EG*, the international endgame study magazine.

The task was simply this. I was presented with three positions known to be won for the side with the rook in a battle against a lone knight, and I was to win them. The diagrams R1, R2 and R3 show the positions, and this is how the play went from each.

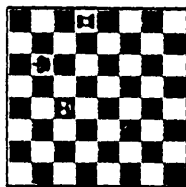
	R1	R2	R3
1.	Rb3	Ka5	Kc5
2.	Kc7	Nf2	Rh7+
3.	Kc6	Na4	Kd6
4.	Rf3	Nd1	Rh8+
5.	Kc5	Nb2	Rh4
6.	Rh3	Nd1	Kd7
7.	Kc4	Nb2+	Kc6
8.	Kc3	Ka3	Rh7+
9.	Rg3	Na4+	Rh6+
10.	Kc4+	Ka2	Re6
11.	Rg5	Nb2+	Kd6
12.	Kc3	Nd1+	Kd7
13.	Kc2	Nc3+	Re3
14.	Kd2	Nc4+	Re4
15.	Kc3	Nc3	Kd6
16.	Re5	Ng4	Kd5
17.	Re6	Kb1	Rf4
18.	Kd2	Nf2	Rf3
19.	Rb6+	Ka2	Kg6
20.	Rb4	Nh1	Rd3
21.	Rg4	Ka3	Kc4
22.	Ke3	Kb3	Kb4
23.	Rg1	'resigns'	Rd1
			'resigns'
			Rb1
			Ke7
			Ra3
			Nc2
			Rg3+
			Kf5
			Ne1
			Kf4
			Ng2+
			Kf3
			Ra3
			Nh4+
			Kf4+
			Kf3
			Ne1+
			Nd3+
			Kc2
			Nf4+
			Kf3
			Nd3
			Rd5
			Nb4
			Kg1
			Rd6
			Nc2
			Kf5
			Rg6+
			Nh2
			Rg4
			Na1
			Rb4
			Kh3
			Kh2

I played with a clock (at the time-limit of 16 moves per hour) and I did not have any moves back. On the other hand, I had had over a week's notice of the positions, and naturally I prepared for the contest. In chess terms it was more like having an adjourned game.

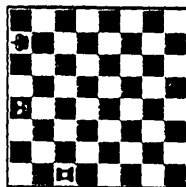
I was already familiar, without being move-perfect, with what the books give on this ending. Also, I had learned a lot from playing against



R1



R2



R3

Three positions, all White to play. In all cases White captures the black knight on his 24th move.

the same database at the April two-day conference in Edinburgh (CW, May 25). Thirdly, I asked for, and was given in advance, four positions (and the single-line optimal play) with the same solution length as the contest positions, so that I could have some training that was relevant (I refer to these later as the Edinburgh lines).

The solutions were all 24 moves long, chosen at random from the 178 such positions, with the idea that the near-maximum length (no position has an optimal solution exceeding 27 moves) and the number of such positions would provide the toughest and fairest contest conditions.

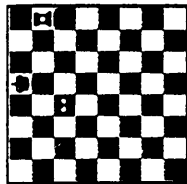
About five minutes was spent on each task. My preparation was pretty good, and most of the difficult moves I had already written down. Not only did I have notes, but an auxiliary board on which I moved the pieces around when I wanted to before choosing the move to be played on the primary board. This was all agreed in advance. But I used the auxiliary board seriously only twice. The contest was between a chess analyst and a database, not between an over-the-board master and a database, so the conditions were deliberately made to ensure the analyst, myself, could not complain.

Coming to the play, the solution to R3 is very similar to that to R1, a mirror image, or almost, from move 5 onwards. We were all surprised at the lack of variety, certainly the apparent lack of variety, in the core of the solutions. Even R2 was like one of the positions given to me in advance, and the R1 and R3 lines were not new to me. But R1 and R2 are quite distinct from each other.

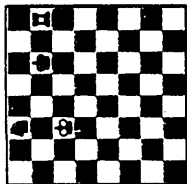
The program might have diverged, by selecting an alternative equal depth move at many possible branch-points, though it did not. But certainly most of the work was in the preparation, and a great deal of the play was accidentally (not deliberately, anyway) to be found in the practice positions.

Speaking as an analyst, R2 is quite beautiful. And it will not be found in the textbooks. Take the position after Black's third move (see R2.1). The black knight and black king are striving to meet, to set up a standard drawing set-up with both men next to each other on the edge. White has to play very precisely to prevent this. In fact, I find the next 10 moves a sheer delight.

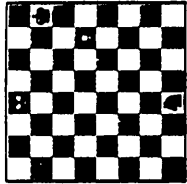
Consider the position after 8 Rh7+ (see R2.2). Where can Black play



R2.1



R2.2



R4

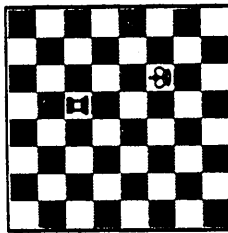
his king? If he goes to e8 or f8, the knight is immediately lost to a rook check on h8. If he goes to g8, however, the rook plays to c7, winning the knight by domination, since it has no safe squares. The same applies to the move of Black's king to g6. So, in spite of the apparently wide choice, Black really has only e6 and f6 to choose from. But even e6 fails to Rh6+ and the variety of ways in which the knight is now caught in just a few more moves is quite delightful. So, really, only f6 will do for Black in R2.2. The effect, in the mind, of comprehending each of these variations individually, and all of the variations as a group, is strongly aesthetic.

After R1, R2 and R3 had been disposed of, I asked for three more positions to be given to me without preparation. Here I would try to play fast, but with up to three moves back — a kind of compromise between over-the-board competitive play and how the analyst works. The result was exactly the same. I made no mistakes (to everyone's surprise, including my own) and it was remarkably easy. I assumed that the positions would transpose into solutions that I already knew — indeed, into the essentially different R1 and R2 lines or into one of the Edinburgh lines. And they did. An example is R4. It was not too difficult to find 1 Ke7 Kh6, 2 Kf6 Ne3, 3 Rg3 Nd5+, 4 Ke6 Nc7, 5 Kf5 Ne8, 6 Rg6+ Kh5, 7 Rc6 and we are in R1, as near as makes no difference. I had no moves back in these three 'unseen' positions. It turned out that all the moves I made were optimal. That is, no moves I made prolonged the solution unnecessarily.

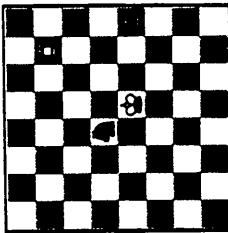
I tackled the contest by looking for patterns. For instance, the play in all these long-solution positions seems to fall naturally into three phases. In the first phase the chessmen are relatively dispersed, but Black must get his king and knight together, or perpetually threaten to. At the end of this we enter a second phase, the kernel of the struggle, where a sequence of a dozen or so moves decides it, because eventually the knight is forced away from its king. Phase three mops up the knight. Of course, mate threats operate in any phase of the struggle, even in all phases.

## Appendix 4. Benchmark of Twenty Positions

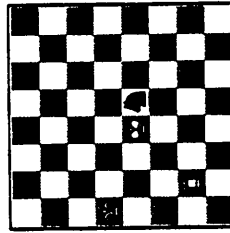
The following pages contain our benchmark of 20 positions which illustrate all the features of special difficulty for the knight's side of the ending of which we are aware. All positions are drawn with correct play.



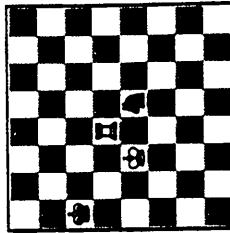
D1 ... Ng8



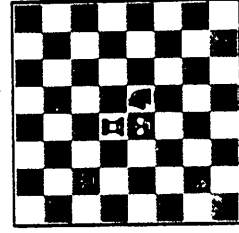
D2 ... Nc3+



D3 ... N46

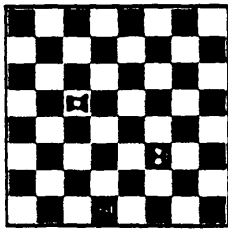


D4 ... Kb7

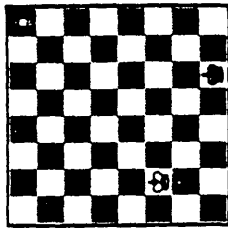


D5 ... N47+

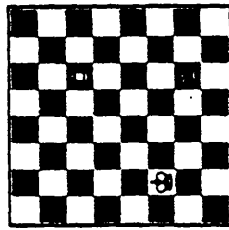




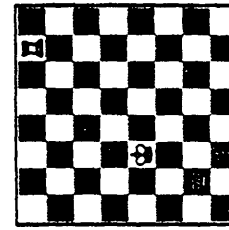
D5.2 ... Nb1+



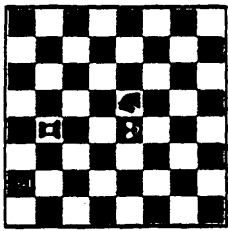
D6 ... Kc6



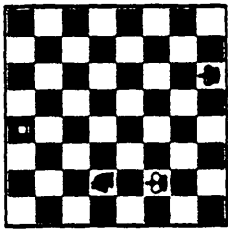
D8 ... Kc6



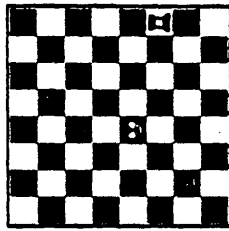
D8.2 ... Kc7



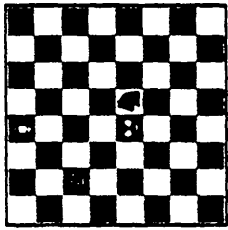
D9 ... Kc6+



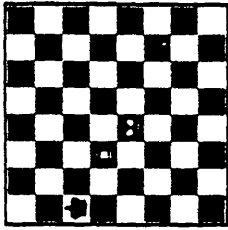
D10 ... Kc6



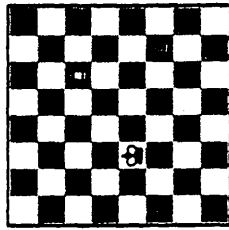
D12



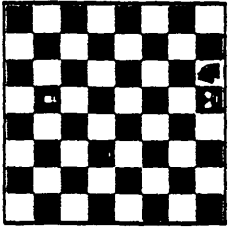
D13 ... Kc7 or Kc2



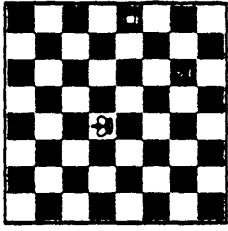
D16 ... Kc6



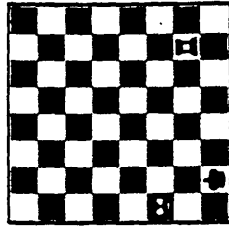
D18 ... Kc4



D15 ... Kc6



D17 ... Kg3



'Corncase' Rb2+ ...