Candidate Relevance Analysis for Selective Search in Shogi

Reijer Grimbergen
Electrotechnical Laboratory,
1-1-4 Umezono, Tsukuba-shi, Ibaraki-ken, Japan 305-8568
E-mail: grimberg@etl.go.jp

Abstract

In this paper we present Candidate Relevance Analysis (CRA), a method for selective candidate generation in two-player perfect information games. Selective candidate generation has not been widely used because of the risks of losing important search candidates. However, there are search domains where the number of search candidates is too large to do a complete search. CRA limits the number of search candidates by analysing the relevance of the candidates with respect to the goal of a game, discarding all candidates judged irrelevant. CRA can easily be incorporated in alpha-beta search. Furthermore, the knowledge acquired by CRA can be used to improve the move ordering, thus improving the efficiency of alpha-beta search. In the domain of shogi (Japanese chess), CRA reduces the number of candidates by 39.7%, while maintaining an accuracy of 99%. Also, in shogi alpha-beta search with CRA outperforms full-width alpha-beta search even for small search depths.

1 Introduction

In the early days of game playing research, selective search was the main method used (Newell *et al.*, 1958; Bernstein and Roberts, 1958; Greenblatt *et al.*, 1967). A plausible move generator would select a number of search candidates using domain-specific knowledge. The remaining candidates were then searched as deep as possible with alpha-beta search. For example, Bernstein's chess program generated only 7 plausible moves in any position. Selective search was considered to be the natural approach because human players also select a limited number of moves to search. Furthermore, hardware was usually too slow to search all candidate moves deep enough for good play.

This changed with the development of the Chess 4.5 program in the early seventies (Slate and Atkin, 1977). Slate describes how they first followed the common selective search method of finding plausible moves but in the end decided to mark all moves as plausible. Therefore, the main algorithm became a full-width alpha-beta search, searching all legal moves to the same depth (called the nominal search depth). Added to this were transposition tables and an evaluation function in which the domain-specific knowledge was concentrated (in this case, the chess knowledge). This method was very successful, with Slate's program dominating computer chess tournaments for most of the 1970s. As a result, full-width alpha-beta search has become the standard search method for game playing programs. Recently, this has led to well-tuned programs that have reached human world champion level performances in checkers (Schaeffer, 1997), othello (Buro, 1997) and chess (Schaeffer and Plaat, 1997).

The risk of discarding a good search candidate or even discarding a perfect solution to a search problem has limited the development of selective search. There are two reasons why we feel selective search deserves more attention. The first reason is the link between selective search and human problem solving. Trained humans are very good at selective search, even though they sometimes make mistakes. To know more about problem solving by human experts, we feel that selective search needs to be investigated further.

The second reason is that there are domains where it is not possible to search all search candidates deep enough to reach expert level performance. This can either be caused by too many search candidates (e.g. Go, shogi) or by solution sequences that are too long (e.g. sokoban). It seems that in these cases, there is no other choice than to give up the completeness of the search.

Within the framework of alpha-beta search there are two general ways by which completeness of the search can be given up: 1) limit the number of search candidates, i.e. *selective candidate generation*, or 2) stop the search before reaching the nominal search depth, i.e. *forward pruning*. An example of selective candidate generation is the use of pattern databases to generate moves in Go (Kojima and Yoshikawa, 1998). Examples of forward pruning are *futility pruning* (Heinz, 1998) and *multi-cut pruning* (Björnsson and Marsland, 1998).

In this paper, we present a method for selective candidate generation for two-player perfect information games, called Candidate Relevance Analysis (CRA). It is based on the analysis of the relevance of a candidate move. If a move has no relevance for reaching the goal of the game, it is not searched. Using this selective candidate generator, normal alpha-beta search can be applied. It will be shown that CRA not only limits the number of search candidates, but also provides knowledge that can be used for ordering the candidate moves, improving the efficiency of alpha-beta search. CRA is explained in Section 2.

In Section 3, CRA is applied to *shogi* (Japanese chess). It will be shown that in shogi considerable savings can be achieved with minimal risk of discarding a good candidate move. Other results show that alpha-beta search with CRA outperforms plain full-width alpha-beta search in shogi. These results will be given in Section 4.

In Section 5 we end with some conclusions and thoughts on further research.

2 Candidate Relevance Analysis

In this section we describe Candidate Relevance Analysis (CRA), a method for selective candidate generation in two-player perfect information games. We will first define how CRA limits the number of search candidates by finding relevant candidate moves in a position, discarding all moves that are not judged relevant. This analysis is based on the relevance of a candidate move for the goal of the game. The knowledge acquired during the analysis of relevance can be used for move ordering. This method can then be integrated in alpha-beta search by replacing the move generator and initial move ordering with CRA.

CRA is closely related to the plausible move generators in chess programs in the 1950s and 1960s. However, because of the slow hardware, these approaches were forced to limit themselves to the generation of a small set of plausible moves. This is a natural approach, since cognitive research has shown that human experts are capable of narrowing the number of search candidates to a very small number of relevant candidate moves (de Groot, 1965). After this, search is carried out to make a decision between the relevant moves.

CRA does the opposite: discard all moves that are not plausible. Even though in theory this leads to the same set of candidate moves, in practice the philosophy is very different. CRA aims at a candidate move set that is as safe as possible, including many candidate moves that will only have relevance in very specific positions. Taking advantage of the improved hardware speed and the enhancements of the alpha-beta algorithm that were not available 30 years ago, it is now possible to search much larger search spaces. In contrast, having a small search space by deciding upon a limited number of plausible candidate moves is much harder because it requires a lot more knowledge. It has been estimated that human expert chess players have at least 50,000 chunks of chess piece configuration knowledge (Simon and Chase, 1973). If the knowledge is insufficient, the chance of losing vital moves becomes too high, leading to poor play. Therefore, it seems more promising to have a simpler analysis which removes

less candidate moves, but removes enough candidates to improve the efficiency of the search.

2.1 Goal Related Relevance

As the general game-playing system METAGAMER (Pell, 1996) shows, a wide range of games have the notion of threat and goal in common. The goal of a game can be to either win material (e.g. chess, checkers), to occupy the largest territory (e.g. Go, othello), or to reach a certain board configuration (e.g. five in a row, sokoban). In most games there are subgoals. For example, in chess the goal of the game is to take the opponent's king. Even if this can not be achieved immediately, taking the opponent's queen might be a very reasonable subgoal to eventually reach the goal. In Go, winning the battle for a small territory will be a subgoal in obtaining the larger overall territory.

Our assumption is that if a move has no relevance with respect to the goal of a game, it can be cut from the search. We call a move *relevant* if it fits into one or more of the following categories:

- Reach (sub)goal
- Threat to reach (sub)goal
- Defend against opponent's threat to reach (sub)goal
- Develop position to improve the chance to reach goal
- Develop position to better prevent the opponent from reaching goal

The last two categories are the hardest to define in a general way because the notion of development usually requires domain-specific knowledge. This can not be helped, since human players also need domain-specific knowledge to improve their strategic awareness of a game (e.g. by playing many games against strong opponents or by the study of expert games). Still, from Pell's work (Pell, 1996) on METAGAMER, it can be concluded that even in these categories there are some features that carry over between games. An example is the general concept of mobility, i.e. keeping as many move options as possible.

2.2 Move Ordering

Move ordering is vital for guiding alpha-beta search. Searching good moves early quickly establishes narrow bounds which speed up the rest of the search. In CRA, the categories for relevance represent a hierarchy that can be used for move ordering. For example, reaching the goal of the game takes priority over all other categories.

This ordering can not be perfect for every position. Usually, defending against the opponent reaching a subgoal has a higher priority than developing the position, but that is not always the case. Moreover, without further analysis the relative priorities of threats and defences against threats can not be decided.

Therefore, the categories given above only give a rough ordering of relevant moves. In general, this is where search has to help out, even though domain-specific knowledge can be used to further improve the ordering, as we will see in Section 3.

Another point is the possibility that a search candidate fits into more than one category. There is a trade-off here. If a candidate is relevant according to multiple categories, then it is likely that this candidate is better than a candidate that is relevant according to a single category. However, analysing the relevance of a candidate for all categories can be time consuming. The decisions on which categories to test depend on characteristics of the domain in which CRA is applied.

3 Application to Shogi

The rules of shogi are similar to chess, with mating the opponent king as the goal of the game. Shogi is played on a 9x9 board with pieces that are different from those used in chess (no queen, but generals and lances). Also, while in chess only pawns can promote to other pieces, in shogi most pieces have promotion abilities. However, the main difference between chess and shogi is the possibility of re-using pieces. A piece captured from the opponent becomes a *piece in hand* and at any move a player can *drop* a captured piece on a vacant square instead of moving a piece on the board. This possibility of re-using pieces has serious consequences for the search complexity of shogi.

First, the number of candidate moves increases dramatically. In chess the average branching factor of the search tree is 35, while in shogi it is about 80 (Matsubara and Handa, 1994). This is especially a problem in the endgame, where the number of legal moves often reaches 150 or more.

Second, quiescence search is much more difficult than, for example, in chess. The possibility of dropping with check or dropping with an attack on a piece makes it harder to find quiet positions that can be evaluated in a reliable way.

Third, static evaluation of positions becomes more difficult because of the dynamic nature of the game. While evaluation functions in most chess programs rely heavily on material balance, in shogi there is not even consensus on the relative value of pieces (Beal and Smith, 1998). The shogi endgame is a race to mate the king of the opponent and in this race material balance is of secondary importance.

As a result of these problems, shogi playing programs still only have a strength of approximately 1-dan, which roughly corresponds to 2000 ELO points in chess.

3.1 Goal Related Relevance in Shogi

The problem of having a large number of candidate moves makes shogi a good test domain for CRA. In shogi, the categories of Section 2.1 can be interpreted as follows:

- **Reach (sub)goal**
  - Take king; Take material; Promote piece
- **Threat to reach (sub)goal**
  - Give check; Attack squares close to the king; Attack pieces of higher value; Threat to promote piece
- **Defend against threat to reach (sub)goal**
  - Defend against check; Defend squares close to the king; Move attacked piece; Defend attacked piece; Defend against opponent's promotion
- **Develop position to improve the chance to reach goal**
  - Improve piece mobility; Standard development of piece; Build common attack formation
- **Develop position to better prevent the opponent from reaching goal**
  - Restrict opponent mobility; Build common defence formation, e.g. by castling

Most of the shogi specific subcategories speak for themselves and can be defined accurately by a small number of rules. Exceptions are moves for the standard development of pieces and moves for building common attack and defence formations. The rules for testing if moves belong to these subcategories have been implemented with patterns of piece configurations to match with the current position. This pattern matching is relatively slow.

3.2 Move Ordering in Shogi

As stated in Section 2.2, the analysis of the relevance of a move can also aid in the ordering of the move candidates. For example, taking a free piece with check is usually better than just checking. Also, if one's king is safe, but the opponent's king is in danger, attacking moves should have a higher priority than defensive moves. For shogi we propose the following order based on the categories above plus some common sense shogi knowledge:

1) Take King
2) Defend against check
3) Win material with check
4) Win material
5) Move attacked piece
6) Promote piece
7) Defend against opponent's promotion
8) Obtain pieces in hand by exchange
9) Multiple attack on higher valued piece
10) Single attack on higher valued piece
11) Move to threaten promotion
12) Defensive move
13) Attacking move
14) Castling move
15) Other position developing move
16) Move piece to safe square
17) Material sacrifice

If a move in any category loses material instantly, it is ordered lowest. This means that our implementation is not capable of recognising good sacrifices. In shogi sacrifices are more common than in chess, but we will see later that they are still relatively rare. Therefore we think that the ordering given here is a good ordering in general.

For the moves that do not lose material instantly, the moves to develop the position have a very low priority. Furthermore, testing if a move fits this category is slow, because a large number of standard patterns of pieces have to be matched with the current position. To speed up the analysis of relevant candidates, a move is not tested for position development if its relevance has already been established by one of the other categories.

We use one context dependent rule to rearrange move ordering based on the current position. If the opponent's king is in danger, attacking moves are moved up to rule 6. Likewise, if the player's own king is in danger, defensive moves are relocated in the same way.

4 Results

We have performed tests for the three aspects of CRA discussed above: selective candidate generation, move ordering and alpha-beta search with CRA. The first test evaluates the reliability of the relevance analysis. The second test assesses the quality of the move ordering proposed in Section 3.2. The third test compares three different search algorithms to see if smaller search trees and better move ordering outweigh the loss of speed caused by extra analysis.

4.1 Candidate Generation Results

We have tested our approach on 80 games between professional shogi players. The games were picked to represent a wide range of shogi positions. Therefore, games with many different players, many different opening strategies and different time limits were chosen.

In each position of these games the relevance of all legal moves was analysed according to the categories of Section 3.1. If a move was not considered relevant by CRA, it was removed from the list of possible candidate moves in that position. After this, it was checked if the actual move played was among the remaining candidates.

The results of selective candidate generation are summarised in Table 1. The 80 games had a total of 9431 positions. The average savings were 39.7%. The best performance in a single game was 52%

Mvno	*#Pos*	*#TotMv (Avg)*	*#GenMv (Avg)*	*Sv(%)*	*NG*
1-10	800	26379 (33.0)	20630 (25.8)	21.8	-
11-20	800	29509 (36.9)	18928 (23.7)	35.9	1
21-30	800	34033 (42.5)	19291 (24.1)	43.3	7
31-40	800	38456 (48.1)	21353 (26.7)	44.5	17
41-50	800	45646 (57.1)	24779 (31.0)	45.7	13
51-60	800	54651 (68.3)	30087 (37.6)	45.0	19
61-70	791	65613 (83.0)	37810 (47.8)	42.4	10
71-80	790	76237 (96.5)	44734 (56.6)	41.3	10
81-90	763	81363 (106.6)	45409 (59.5)	44.2	3
91-100	644	70906 (110.1)	44503 (69.1)	37.2	4
101-110	517	57505 (111.2)	37038 (71.6)	35.6	6
111-120	407	47623 (117.0)	30472 (74.9)	36.0	1
121-130	267	33148 (124.1)	22002 (82.4)	33.6	3
131-140	185	20327 (109.9)	12933 (69.9)	36.4	-
141-150	100	12394 (123.9)	8039 (80.4)	34.7	1
151-160	67	8432 (125.9)	5267 (78.6)	37.5	-
161-170	42	4956 (118.0)	2984 (71.1)	39.8	-
171-180	20	2667 (133.3)	1609 (80.5)	39.7	-
181-190	10	2256 (225.6)	1375 (137.5)	39.1	-
191-200	10	2271 (227.1)	1449 (144.9)	36.2	-
201-210	10	2014 (201.4)	1110 (111.0)	44.9	1
211-220	8	1783 (222.9)	896 (112.0)	49.8	-
Total	9431	718169 (76.1)	432698 (45.9)	39.7	96

Table 1: Savings and accuracy results in 80 test games. *Mvno* is the move range of the analysis, *Pos* is the number of positions in this move range, *TotMv* is the total number of legal moves, *GenMv* is the total number of moves generated by relevance analysis, *Sv* is the percentage of savings, *NG* is the number of moves played by the expert but discarded by the relevance analysis.

discarded moves by CRA. The worst case was a game with only 27% discarded moves. In Table 1 we can see that the savings are low in the opening of the game at 21.8%, then improve quickly to over 40% in the middle game and then drop slowly toward the endgame. This drop of savings in the endgame might be a problem, since the average number of candidate moves rises continuously. Except for the first ten moves, there is no stage of the game where the savings are far from the average percentage.

In 96 of 9431 positions the move chosen by the human expert was not judged relevant by CRA (1.0%). In 25 games all moves played by the professionals were considered relevant. The maximum number of moves chosen by the expert but discarded by CRA in a single game was 6 moves (this happened in two games). The average game length of the 80 games was 118 moves. Dividing this into opening, middle game and endgame, the endgame will on average start between the 70th and 80th move. There it is vital that relevance analysis is accurate, since missing a good move in the endgame can mean the difference between winning and losing. From Table 1 we can see that starting from move 71, only 29 moves were missed (30% of the total number of missed moves).

4.2 Move Ordering Results

As a second test, we looked at the results of the move ordering proposed in Section 3.2 on the same 80 professional games. For each game position, we investigated where the move played by the human expert ended up in the ordering. The results of this test are given in Table 2. Here we can see that

Ordering	*Abs.*	*Perc.*	*Cumul.*
1	1984	21.0%	21.0%
2-5	2724	28.9%	49.9%
6-10	1360	14.4%	64.3%
11-15	1021	10.8%	75.1%
16-20	685	7.3%	82.4%
21-	1561	16.6%	99.0%
Not Generated	96	1.0%	100.0%

Table 2: Move ordering results in 80 test games

about 50% of the moves played by human experts are ordered among the best five moves according to our move ordering. This increases to almost 65% when examining the ten best moves in the ordering. There is still room for improvement here, with almost 17% of the expert moves ordered lower than twenty. This number seems to indicate that the number of sacrifices in shogi is quite large, since we order moves that immediately lose material lowest. However, when we analysed the moves that were ordered low, we found that less than 10% of these moves were material sacrifices.

4.3 Comparison with Full-width Alpha-beta Search

To final test we performed was to get an indication if the extra time spent on CRA is worth the effort. To do so, we compared three types of alpha-beta search. The first is a very simple full-width search without any move ordering, which we will call *Simple Alpha-beta*.

The second type is a full-width alpha-beta search with the following common sense ordering of moves:

1) Take King
2) Defend against check
3) Win material
4) Promote piece
5) Exchange pieces
6) Move piece to safe square
7) Material sacrifice

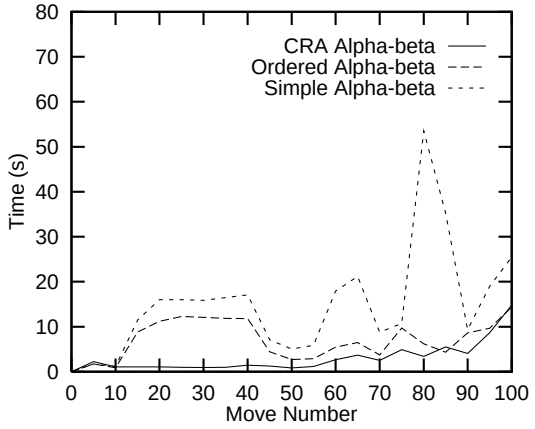

Figure 1: 3 ply alpha-beta search comparison

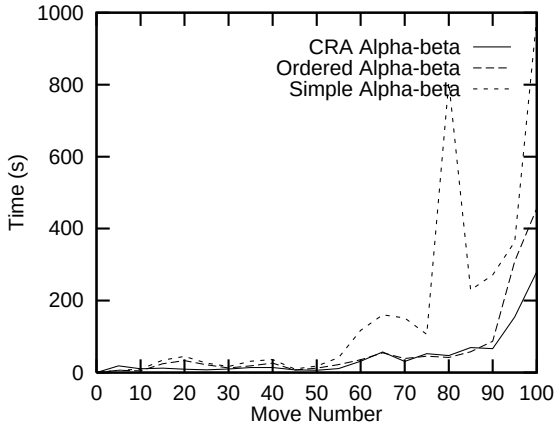

Figure 2: 4 ply alpha-beta search comparison

We call this alpha-beta search *Ordered Alpha-beta*. These two full-width search algorithms were compared to alpha-beta search with CRA (called *CRA Alpha-beta*). This was done by searching 100 positions, being one complete professional game. In all three cases the same evaluation function was used. We compared the performance of the three algorithms for a nominal search depth of 3, 4 and 5 ply.

The results are given in Figures 1–3. On the X-axis the move number is given, on the Y-axis the time that the search needed to finish is given in seconds. In Figure 1 we can see that CRA Alpha-beta is better than the other two types of search for most of the game, giving especially good results in the middle game between the 10th and 50th move. CRA Alpha-beta is more efficient than Simple Alpha-beta in 84 positions and more efficient than Ordered Alpha-beta in 72 positions. As can be seen in Figure 2, the results for 4 ply search are much less clear. Even though CRA Alpha-beta outperforms Ordered Alpha-beta in 60 positions, from the figure it can be concluded that in general these speed-ups are only marginal. CRA Alpha-beta is still much better than Simple Alpha-beta (more efficient in 78 positions). The clearest evidence of the performance of CRA Alpha-beta can be seen in Figure 3. CRA Alpha-beta is better than Ordered Alpha-beta in 88 positions and better than Simple Alpha-beta in 94 positions. Moreover, from the figure it can be concluded that it is a major improvement for all stages of the game except for the first ten moves.

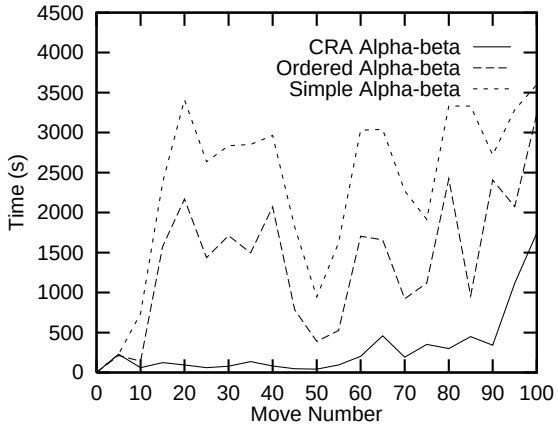

Figure 3: 5 ply alpha-beta search comparison

5 Conclusions

In this paper we have presented a method for selective candidate generation in two-player perfect information games called Candidate Relevance Analysis (CRA). It is based on the analysis of the relevance of moves with regard to the goal of the game in question. If the relevance of a move can not be established, the move is not considered a candidate for search. CRA also gives a method for ordering moves, which can help the search if CRA is incorporated in a search algorithm like alpha-beta search.

We have shown that CRA works well for the game of shogi, giving considerable savings with high accuracy. Even though CRA discards about one move per game on average, most of these moves are opening and middle game moves where good alternatives are available. Further research is needed to establish if CRA as it is presented here is good enough for expert level performance or if improvements are necessary.

We have also shown that CRA improves the move ordering. However, the move ordering provided by CRA has room for improvement and this is also a future work.

Finally, it seems that using CRA is worth the extra effort. Our results show that the plain version of CRA alpha-beta search outperforms the plain version of full-width alpha-beta search, even for small search depths. However, in current game programming there is no program using this simple alpha-beta search. Iterative deepening, the killer heuristic, null move window search and other search enhancements are used to improve the quality and speed of the search. In the near future, we will build an alpha-beta search algorithm with all the enhancements and compare alpha-beta search with CRA and without CRA. Then, we will also play tournaments between different versions of the search algorithms to get a good indication of their relative strengths.

References

D. Beal and M. Smith. First results from using temporal difference learning in shogi. In *Proceedings of the First International Conference on Computers and Games (CG'98) in Tsukuba, Japan*, 1998. To appear in Springer-Verlag's *Lecture Notes in Computer Science* series.

A. Bernstein and M. de V. Roberts. Computer v chess-player. *Scientific American*, 198:96–105, 1958.

Y. Björnsson and T. Marsland. Multi-cut pruning in alpha-beta search. In *Proceedings of the First International Conference on Computers and Games (CG'98) in Tsukuba, Japan*, 1998. To appear in Springer-Verlag's *Lecture Notes in Computer Science* series.

M. Buro. The othello match of the year: Takeshi murakami vs. logistello. *ICCA Journal*, 20(3):189–193, September 1997.

A.D. de Groot. *Thought and Choice in Chess.* The Hague, The Netherlands: Mouton & Co, 1965.

R. Greenblatt, D. Eastlake III, and S. Crocker. The greenblatt chess program. In *Proceedings of the Fall Joint Computer Conference*, pages 801–810, 1967.

E. Heinz. Extended futility pruning. *ICCA Journal*, 21(2):75–83, June 1998.

T. Kojima and A. Yoshikawa. A two-step model of pattern acquisition: Application to tsume-go. In *Proceedings of the First International Conference on Computers and Games (CG'98) in Tsukuba, Japan*, 1998. To appear in Springer-Verlag's *Lecture Notes in Computer Science* series.

H. Matsubara and K. Handa. Some properties of shogi as a game. *Proceedings of Artificial Intelligence*, 96(3):21–30, 1994. (In Japanese).

A. Newell, C. Shaw, and H. Simon. Chess playing programs and the problem of complexity. *IBM Journal of Research and Development*, 2:320–335, 1958.

B. Pell. A strategic metagame player for general chess-like games. *Computational Intelligence*, 12(2):177–198, 1996.

J. Schaeffer and A. Plaat. Kasparov versus deep blue: The rematch. *ICCA Journal*, 20(2):95–101, June 1997.

J. Schaeffer. *One Jump Ahead: Challenging Human Supremacy in Checkers*. Springer-Verlag New York, Inc., 1997. ISBN 0-387-94930-5.

H. Simon and W. Chase. Skill in chess. *American Scientist*, 61:394–403, 1973.

D. Slate and L. Atkin. Chess 4.5: The northwestern university chess program. In P. Rey, editor, *Chess Skill in Man and Machine*. Springer Verlag, New York, 1977.