A Hypothesis on the Divergence of AI Research

Franz-Günter Winkler
Austrian Society for Cognitive Science
Barichgasse 10/2/10, A-1010 Wien, Austria
E-mail: winkler@coams.atc.co.at

Johannes Fürnkranz*
Carnegie Mellon University
School of Computer Science
5000 Forbes Avenue
Pittsburgh, PA 15213
E-mail: juffi@cs.cmu.edu

Abstract

Artificial Intelligence has been conceived as the science of both programming computers to perform intelligent tasks and devising computational models of human reasoning. Originally, both aspects were considered to go hand in hand, but it soon became apparent that AI research is determined to split into a cognitive branch and an engineering branch which correspond to these two objectives respectively. Research in computer chess is a prominent and most successful example for this development. We conjecture that the reason for this divergence is a non-linear interaction between associative human knowledge structures on the one hand and the strict algorithmic processing requirements of computational models on the other hand.

1 Introduction

> *"If one could devise a successful chess machine, one would seem to have penetrated to the core of human intellectual endeavour."*
> (Newell, Shaw & Simon, 1958)

In its early days, Artificial Intelligence (AI) had the two-fold goal of, on the one hand, providing computers with the ability of solving tasks that are commonly perceived as requiring intelligence, and, on the other hand, furthering our understanding of human thinking. The above quotation by Newell, Shaw

*On the leave from the Austrian Research Institute for Artificial Intelligence, Schottengasse 3, A-1010 Wien, Austria

and Simon exemplifies this view: On the one hand we want to achieve *intelligent functionality* ("... devise a successful chess machine ..."), while, on the other hand, we aim at *understanding* the processes that guide *human reasoning* ("... penetrating to the core of human intellectual endeavour.").

Furthermore, it was believed that both aspects of AI go hand in hand: A better understanding of the human intellect is facilitated by the ability to test computational models of cognitive theories, and, conversely, the achievement of intelligent functionality is impossible without a deeper study of how we human beings achieve that functionality. Hence, Newell, Shaw and Simon were convinced that a successful construction of a computer that excels in chess, commonly perceived as a task whose mastery requires intelligence, would have a strong impact on Artificial Intelligence in general.

Now, that the DEEP BLUE team has devised a successful chess machine, it is apparent that, although significant progress has been made in both, our understanding of human cognition and the implementation of intelligent machines, we are not much closer to the original goal of AI, of implementing a genuine artificial intelligence on a computer. Instead, research in AI has diverged into two independent research areas: On the one hand, the what one might call *cognitive* branch of AI (nowadays usually referred to as *Cognitive Science*) puts a strong emphasis on the psychological validity of computational models, in particular with respect to knowledge representation and memory organization. On the other hand, we have the *engineering* branch of AI, which is motivated by solving particular tasks, and is mostly concerned with finding formalizations and software architectures that are tailored to the solution of a specific problem and can be efficiently executed on computer hardware. DEEP BLUE is a most prominent example for this line of research.

In this paper, we illustrate this split of AI into two different branches by classifying research in the domain of chess along two different axes: *human-compatible knowledge (HCK)* and *machine-compatible processing (MCP)*. We further argue on the example domain of chess that the successes of AI research can be found along the two axes, but have not yet penetrated into the white area which we consider to contain the core problems of AI. We then ask the question why this is the case, and present a hypothesis to resolve this issue.

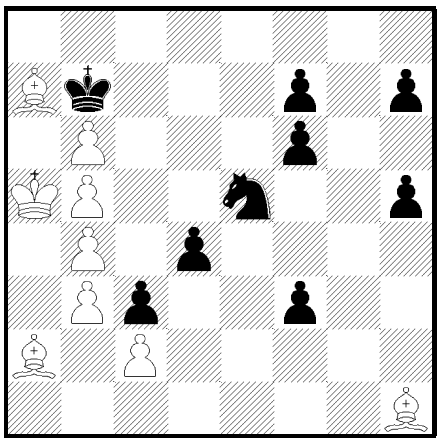
2 Two Illustrative Examples

The basic problem that AI has to face for achieving its original goal of finding computational models of human cognition is to integrate the abstract concepts that human beings usually use for problem solving with the strict algorithmic processing definitions that are required for implementing programs on a computer. To illustrate these dimensions consider the two problems depicted in figure 1.

The first problem is a slightly unusual endgame position, but, after some deliberation, a trained chess player will detect many concepts that are familiar from other endgames. For example, it will soon become obvious that black

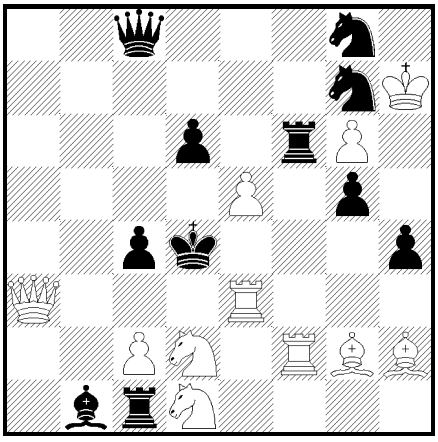
has a dangerous threat with 1...d3, which will either queen the d-pawn or distract white's c-pawn, thus allowing black to queen his own c-pawn. Because of the above drawing chance for black, white's only hope lies in queening one of its pawns on the b-file (after playing 1.Bb1 to prevent 1...d3). To achieve this, white has to conquer the square a6. However, he has no moves that put black into zugzwang, because black can answer all white king moves with king moves b7-a8-b7. A typical maneuver in such position is the so-called *triangle maneuver*, where one king is able to use a 3-cycle to return to its original square, while the other king is only able to make a 2-cycle. White can therefore try to move its king to e1, playing e1-f2-f1-e1, which would gain one move. When the white king returns to a5, we will have exactly the same position, but with black to move. As he cannot move his king because of white's threat Ka6, he has to move one of his pawns. Then the entire sequence is repeated 11 times until black has no more pawn moves and has to answer 254. Ka5 with Kc8 thus allowing Ka6, followed by a mate in 15.

Nenad Petrović, 1969



Mate in 270 moves

Sam Loyd, 1892



Mate in 2 moves

Figure 1: A knowledge-rich and a knowledge-poor problem.

Trained chess players could solve this problem in their heads (maybe with the exception of the final mate in 15). However, because of their lack of high-level chess-specific concepts and general problem-solving knowledge, this problem is very hard for current computer chess programs, as the solution is too deep to be found with exhaustive search. A brief experiment that we have conducted with FRITZ4 has demonstrated that the program will play Bb1 to prevent black from playing d4-d3, bring its king to a1, Ba2, king to e1, Bb1, and finally move the king to f2 in order to capture the pawn on f3. It does not realize that black then has an easy draw by trading its knight for the bishop and playing f6-f5 thereafter.

Conversely, the second position is quite hard for human chess players, because it does not offer many familiar patterns which could be used for narrowing down the search. The pieces seem to be randomly scattered around the board, thus offering no orientation at all. The only obvious feature is the lack of king safety, so that white should be much better off and probably has a mate in a few moves. However, the restriction to look for a mate in 2 proves to be very hard for a player untrained in solving chess problems, because of the lack of familiar patterns that allow to cut down the number of candidate moves in a reasonable way. She would more or less have to perform an exhaustive search through the numerous possible threats, a procedure for which human memory organisation is not particularly well-suited.[1]

On the other hand, positions like these are no different to other positions for a computer chess playing program, and thus it would find the mate as easily as it would find other mate-in-2's, as e.g. the queen sacrifice followed by a smothered mate that every experienced chess player is familiar with. The reason for this is that these algorithms are not modeled after human memory and thus do not have to rely on the recognition of familiar concepts on the board as human chess players do. Instead they rely on state-of-the-art search algorithms that are tailored to be efficiently executable on computer hardware.

We believe that these examples illustrate fairly well the two different aspects of problem solving in which humans and machine respectively excel: On the one hand there is a rich associative memory that is very hard to formalize, while on the other hand there is fast processing of low-level concepts, which is hard to explain in an intuitive way. In the next sections, we will discuss these two aspects in more detail, before we propose a hypothesis on why it is so hard to reconcile them.

3 Human-compatible knowledge

Chess is probably the game that has been most deeply investigated from a theoretical point of view. Chess books are full of comprehensible knowledge about different aspects of the game. We would like to call such knowledge *human-compatible*, because it enables a chess student to increase his understanding and competence of the game. Nevertheless, it is mostly unclear how the student uses this knowledge for problem-solving. Human subjects are often able to specify the abstract concepts they use for problem-solving, but are unable to specify the problem-solving process in an exact algorithmic way. For example, a chess player has no problems in explaining the reasoning that made him prefer a certain move over other possible continuations. Analyses like "The move b4

[1]Note that a player with experience in solving chess problems, such as the anonymous reviewer who pointed this out to us, would face much less difficulties in solving this problem, because he can rely on appropriate high-level concepts: the queen appears to be underused, so we should look for a mating pattern with the queen (like on e3), and the move Rg3 creates a flight, so this is a good first try. Again, it is important to note that the solution has been discovered by the use of abstract concepts that are hard to formalize, as we will discuss in the following.

gives me a backward pawn on c3, but it prevents a black liberation with a5, so that I can attack his weak a6-pawn on the half-open a-file." are full of abstract concepts like *backward pawn*, *half-open file*, etc. that are well-understood by human players. However, it is comparatively difficult for human players to specify the thought processes that made them prefer, e.g., the opponent's weak a6-pawn over their own backward pawn on c3.

Research in chess psychology [7, 4, 17, 8] has extensively analyzed verbal thinking-aloud protocols of chess players of different strengths. The results are that differences in playing strength between experts and novices are not so much due to differences in the ability to calculate long move sequences, but to the use of a library of chess patterns and accompanying moves and plans that helps them choose the right moves for deeper investigations. Several authors have even tried to measure the magnitude of this pattern library, resulting in estimates in the range of 5,000 to 10,000 patterns [37, 16]. Some of these so-called *chunks*[2] are easy to articulate and common to most chess players (like, e.g., *passed pawn*, *skewer*, *minority attack*), while others are presumably subconscious and subjective to individual players. However, even simple concepts like a *knight-fork* are non-trivial to formalize.[3]

Because of this strong focus on models for memory organization, early AI research has concentrated on the simulation of aspects of the problem-solving process that are closely related to memory, like perception [36] or retrieval [38]. Recently, these ideas were re-investigated and integrated into the CHREST program [13], which is the most advanced computational model of a chess player's memory organization. CHUMP is a variant of this program that is actually able to play a game by retrieving moves that it has previously associated to certain chunks in the program's pattern memory [14].

4 Machine-compatible Processing

AI has soon recognized the difficulty of formalizing human thought in a top-down way (using the human concepts as a starting point), and has instead discovered approaches to solving intelligent tasks which are more closely designed to fit the processing requirements of a computer. Brute-force chess programs are the best-known example of this line of research. The basic idea of brute-force chess programs dates back to [35] and [40], where one can already find many of the ideas that are still used in today's chess programs (like, e.g., search extensions). However, early chess programs (see [28] for an overview) relied on highly selective search that evaluated positions based on a few basic concepts like *material balance*, *center control*, and *king safety*. This selective search was

[2] Recent research has extended the chunking theory with so-called *templates* [15], i.e., long-term-memory structures that are quite similar to *scripts* and *frames*, but are based on a detailed psychological model [8]. For our discussion, the differences in the details of the psychological models of chunks and templates are irrelevant.

[3] The basic pattern for a fork is a protected knight threatening two higher-valued pieces, like, e.g., rook and queen. However, this simple pattern might not work if the forking knight is pinned. But then again, maybe the knight can give a discovered check...

motivated by both hardware limitations and the attempt to model machine chess playing after human chess playing.

However, the somewhat unexpected success of the TECH program [12] for the first time demonstrated the power of brute-force computing. Further improvements on the search algorithms and evaluation functions [39] and advances in parallel processing [19] and chess-specific hardware [6, 10, 18] have eventually lead to the DEEP BLUE vs. Kasparov challenge, which resulted in the long-awaited defeat of the human world chess champion in a match in 1997, after he had lost the first game under tournament conditions in a previous match in 1996.

5 Two Dimensions of AI Models

The success of brute-force programs is attributable to the fact that their basic architecture is adapted to what computers are good at: fast calculation using only a few isolated chess concepts, which can be evaluated efficiently. Thus the success of these programs depends on *machine-compatible processing (MCP)*. On the other hand, we have seen that human chess players calculate relatively few moves, but rely on a huge pattern library that helps them select the right move. Thus, their success depends on the availability of *human-compatible knowledge (HCK)*. In the following, we attempt to classify a few AI models in the domain of chess with respect to their contribution along either axis.

A perfect chess program that has access to the perfect game-theoretic values of each position (e.g., by exhaustive search until check-mate) would be on the right end of the MCP axis with no contribution on the HCK axis, as its internal knowledge representation does not contribute in any way to a better understanding of human reasoning. The best-known approximation of this principle is the DEEP BLUE chess program. The other extreme would be an oracle that could derive the best move in each position from general principles and explain this choice in a clear and understandable form. Chess theory can be viewed as an attempt to approximate this knowledge and Garry Kasparov, in some sense, can be viewed as a machine that embodies this knowledge. Thus he appears high up on the HCK axis, with almost no contribution along the MCP axis.

A project like CHUMP (see above) is strongly motivated by human memory organization, and its processing is not very compatible with typical computer hardware. For example, the program uses artificial simulations of human long-term and short-term memory. Therefore, it has only made a small step along the MCP axis. Its contribution along the HCK axis is higher, but it is clearly a simplification compared to human memory organization. As a sort of dual example, consider PARADISE [41], which is a program for solving chess combinations at an abstract level. The main goal of this project was to investigate the extent to which tree search can be guided and controlled with the use of background knowledge [42]. The used concepts are quite abstract and clearly motivated by human knowledge, but the processing is still very machine-compatible, using a systematic best-first search in the space of possible plans.

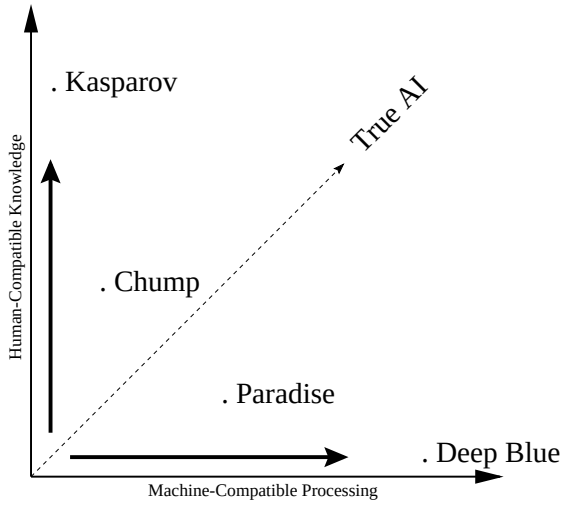


Figure 2: Models of chess playing classified along the two dimensions of human-compatible knowledge (HCK) and machine-compatible processing (MCP).

As we have elaborated above, we view Artificial Intelligence as a science that has to integrate both aspects: human-compatible knowledge and machine-compatible process definitions. Hence, an AI project can be evaluated by its contribution along either axis. Figure 2 positions the "chess models" discussed in the previous paragraphs in this two-dimensional field. We view the overall *value* of a model as the sum of the contributions that it makes along either axis. Intuitively, for a chess program, this value measures the program's competence in terms of playing strength and explanatory power.

A research program appears as a line progressing through this two-dimensional space. The ideal AI research program should follow the diagonal between the two axes, thus taking into account both aspects equally. The further a research program deviates from this diagonal, the less we are inclined to call it research in AI. Clearly, the progress in chess theory (which moves along the HCK axis) can hardly be regarded as part of AI, because of its lack of MCP. Likewise, we think that a certain minimum amount of human-compatible knowledge is required for a computer program to qualify as AI. Along this dimension, the development that has led to Deep Blue, in our opinion, is at the lower border of AI research, if not beyond.

The example of chess shows us that research has split into two streams: One that proceeds close to the HCK axis and is concerned with a deeper understanding of human problem solving and the domain knowledge on which it relies. The other stream is concerned with the development of faster and better algorithms and hardware heading for an exhaustive search. The little work that has been done in the white area inbetween has only been moderately successful in terms

of our combined measure. We believe that this divergence can also be found in many other areas of AI like, e.g., Machine Learning or Natural Language Understanding.

Apparently, progress along the axes faces less resistance than progress along the diagonal. The question is "Why?".

6 A Hypothesis on Research Effort

We interpret figure 2 in the way that moving towards a better model (in terms of the overall value defined above) requires more effort if this improvement in value is made along the diagonal than if one proceeds along one of the axes. We think that the effort that is necessary to proceed a step along one axis is proportional to the progress that has already been made along the other axis. Only in the special case of a research program that proceeds along a single dimension (as e.g. the research in computer chess) will the increase in effort be proportional to the increase in value. This leads us to the following hypothesis:

> *The total effort that has to be spent on an AI problem is proportional to the product of the values along the axes HCK and MCP.*

As an illustration consider the curve depicted in figure 3, which shows a line of AI problems which require the same effort, i.e., problems for which the area of the rectangle spanned by the co-ordinates is constant. We believe that the status quo of AI research could be described with such a curve. Relatively few progress has been made along the diagonal, while considerable progress has been made along the axes. On the other hand, consider the line which depicts a class of AI models sharing the same overall value. According to our hypothesis, there is a noticeable discrepancy between these two graphs, i.e., there is a difference between the value of an AI model and its required effort.[4]

From these deliberations, it follows that HCK and MCP cannot be treated independently. A possible way how they are interrelated can be derived within the theory of semantic networks. This theory seems adequate for this task, because, on the one hand, semantic networks are a psychological model of human memory organization and, on the other hand, several variants have been successfully implemented and had a considerable influence on AI research.

A semantic network consists of nodes and links that connect such nodes. The nodes are to be understood as representations of basic concepts like objects or properties, whereas the links represent relations between these concepts. The

[4]Note that this curve is not equivalent to the well-known search vs. knowledge trade-off studied, e.g., in [2] and [20]. For example, perfect endgame databases are used in many playing programs. They provide a maximum amount of knowledge with no search. However, the knowledge contained in these databases is not very human-compatible, as can be seen from the hard time that various endgame specialists had in understanding and extracting playing strategies from such databases [34, 30]. On the other hand, they are very machine-compatible: a simple database lookup suffices for perfect play. Hence, we would plot such approaches with low HCK, but high MCP values, at about the same spot where we would put programs that play such endgames using brute-force search.

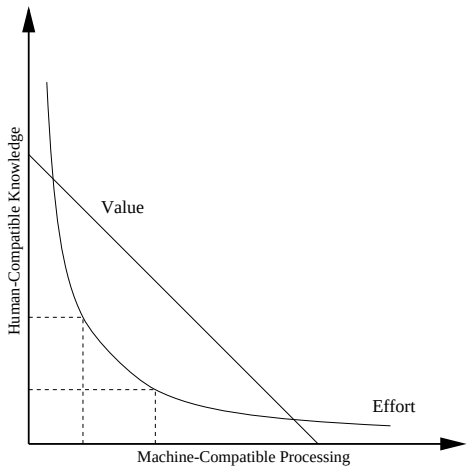


Figure 3: The discrepancy between a class of models that are equal in effort and a class of models that are equal in value.

resulting structure is a model of human semantic memory organization. Processing in such a semantic network is guided by the links (either in the form of explicit rules or by spreading activation, i.e., a sufficiently activated node spreads some of its activations to its neighbors, which in turn may receive enough activation to spread out). When there are sparse links between nodes that form a linear and mostly hierarchical structure, computation will be quick and easy. When the link structure is dense and cyclic, computation becomes problematic due to circularity and to the exponential growth in the number of possible paths. Thus processing depends on the number and the structure of the links in the network.

Classical semantic memory models [5, 1] rely on the power of hierarchical, sparse network structures, but there is strong critique on this situation. In [24], Klimesch argues that these conceptions are misleading and he shows that they contradict important experimental data. He instead pleads for highly interconnected and cyclic memory structures in the following sense: The more connections a concept is associated with, the more meaningful is that concept. The expert who really understands what he is doing and who can apply his knowledge in most different situations has a much more interconnected memory structure than the beginner who just learned some strict rules. The interesting thing is that human memory performance becomes better and quicker with increasing connectivity of concepts, which is exactly the reverse for computational processes.[5]

[5]One of the authors made extensive experience with this and other facets of problematic computability of interconnected structures when simulating Klimesch's model in his diploma thesis [43].

While our discussion on the trade-off between HCK and MCP has been mostly qualitative, we believe that in the formalism of semantic networks it is possible to define operational measures for HCK and MCP on the basis of the network topology: HCK is high when the link-to-node ratio is high as in the interconnected and cyclic structures of human semantic memory. On the contrary, MCP is high when the node-to-link ratio is high as it would be the case in sparse and linear structures that are suitable for computer processing. Such a construction makes the interrelation of the two axes explicit and directly supports the product law we postulated.

7 Steps towards a Re-unification of AI

We take the quotation preceding our paper as an intuitive definition of the goal of AI. The goals expressed in this statement are on the one hand to produce intelligent behavior in the form of a "successful chess machine" and, on the other hand, to "penetrate to the core of human intellectual endeavor". However, the development of AI research, in particular in the domain of chess, has shown that the latter is not an immediate consequence of the former. Achieving functionality does not necessarily increase the understanding of how *we* achieve this functionality. We think the original motivation of research in computer chess, namely "merely" to build a successful chess machine, has to be replaced with different goals that require a reconciliation of machine-compatible processing with human-compatible knowledge. However, according to our hypothesis, we cannot expect this to be easy (if possible at all). In the following we would like to give a few examples for rewarding and challenging tasks in the domain of computer chess.

A very rewarding task would be the development of a computable vocabulary of chess concepts in which chess knowledge can be formulated. The characteristics such a representation formalism has to incorporate are that it has to be sufficiently expressive for formulating abstract strategic concepts, that it has to be extensible and can be easily understood by a user (HCK), and that it can be efficiently implemented (MCP). The need for such formalisms has been recognized early in computer chess research. [44] describes an advice-taking chess program which aimed at allowing a chess master to "advice" a playing program in terms of this language. Many formalisms have subsequently been developed in the same spirit [3, 11], most of them limited to certain endgames (see [26] for a bibliography). A recent promising step into the right direction can be found in [9], which introduces a very efficient interpreter of an extensible language for expressing certain characteristics of a board position. However, the expressiveness of the language is currently limited to propositional logic, a trade-off that had to be made because of efficiency considerations and the ability to provide a graphical interface that also allows untrained users to formulate rules.

Another promising field for further research could be the discovery of understandable knowledge in chess endgame databases with the goal of enriching chess theory. Consider, for example, Ken Thompson's impressive work on five-men

endgame databases, which is now publicly available on three CD-ROMs. The use of these disks allow chess programs to perfectly play the encoded endgames. However, many of these endgame databases are not thoroughly understood by human experts. The most famous example are the attempts of grandmasters to defeat a perfect KQKR database within 50 moves or the attempt of an endgame specialist to defeat a perfect database in the "almost undocumented and very difficult" KBBKN endgame [34]. GM John Nunn's effort to manually extract some of the knowledge that is implicitly contained in these databases resulted in a series of widely acknowledged endgame books [29, 31, 32], but Nunn readily admitted that he does not yet understand all aspects of the databases he analyzed [30]. It would be rewarding to develop algorithms for automatically discovering playing strategies for such endgames (see [27] for some preliminary work). A particularly hard problem is that human-compatible strategies are typically simple, but not necessarily *optimal* in the sense that they require a minimum number of moves. For a machine, it is non-trivial to decide which suboptimal moves contribute to some global progress (and are thus part of a useful strategy) and which suboptimal moves do not improve the position. An attempt to automatically discover a simple playing strategy for the KRK endgame might easily produce the simple strategy "always move your rook away from the enemy king" which will always result in a won position (at least for the next 49 moves), but clearly make no progress towards the goal of mating the opponent's king. Other tasks that could be automatized include the discovery of opening theory mistakes, the automatic detection of particularly promising or unpromising line-ups or middle-game plans in certain types of openings, and many more. One can even imagine facilities that support tournament preparation by analyzing game databases with the aim of unearthing characteristics of the style of individual players and for studying their weaknesses and strengths.

Another obvious point, where chess knowledge would be of considerable importance, and probably the point with the highest commercial potential is the use of high-level chess knowledge in educational chess programs. For example, imagine a program that analyzes a certain position or an entire game on an abstract strategic level, tries to understand your opponent's and your own plans, and provides suggestions on alternative ways to proceed. Some commercial programs already provide such capabilities, but at a very preliminary level that usually is only able to detect tactical, but not strategic mistakes. The ICCA has recognized the potential of such programs, and has created the *The Best Annotation Award* which will be awarded annually for the best computer-generated annotation of a chess game.[6] However, the competition suffers from a considerable lack of participants. Some preliminary work on using case-based reasoning for a strategic analysis of a given chess position can be found in [22, 23].

Last but not least, we also believe that additional knowledge can increase the playing strength of current chess programs. However, the motivation to investigate such approaches has significantly declined with the somewhat unexpected success of brute-force programs. We have already illustrated the weakness of

[6] See *ICCA Journal* 15(4):235–236, 1992.

brute-force chess programs in certain endgame positions that require abstract problem-solving and chess-specific knowledge. For some preliminary ideas incorporating strategic long-term knowledge into conventional chess programs see [21, 33, 9]. However, we are also of the opinion that some of the early approaches to selective search need a re-evaluation in the light of the development of AI in the last 25 years.

8 Conclusion

In this paper, we described research in AI, in particular in the chess domain, along two axes, human-compatible knowledge and machine-compatible processing. In this framework it became apparent that AI research has diverged into two streams that proceed along these axes, while we believe that the core of AI lies along the diagonal. For us, the reason for this development is that the effort for combining both aspects is considerably larger than for one-dimensional endeavors, and we have offered a hypothesis on research effort that could explain this observation. It follows that these aspects cannot be treated independently, if one wants to make progress along both dimensions.

The development that led to the DEEP BLUE vs. Kasparov matches, to the first game that a machine won against the human chess world champion in 1996, and finally to its first match win in 1997, in our opinion, demonstrates that proceeding along the "engineering" axis only, hard as it certainly has been, is comparatively easy. While Computer Chess has played a pioneering role in demonstrating that intelligent functionality can be achieved without a deeper understanding of how humans achieve that functionality, we think that it is time for moving towards a re-integration of the cognitive and engineering branches of AI and that computer chess provides a rewarding set of challenging problems that lead into that direction.

References

[1] J. R. Anderson. *Language, Memory, and Thought.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1976.

[2] Hans Berliner, G. Goetsch, Murray Campbell, and Carl Ebeling. Measuring the performance potential of chess programs. *Artificial Intelligence*, 43:7–21, 1990.

[3] Ivan Bratko and Donald Michie. A representation of pattern-knowledge in chess endgames. In M.R.B. Clarke, editor, *Advances in Computer Chess 2*, pages 31–54. Edinburgh University Press, 1980.

[4] William G. Chase and Herbert A. Simon. The mind's eye in chess. In W.G. Chase, editor, *Visual Information Processing: Proceedings of the 8th Annual Carnegie Psychology Symposium.* Academic Press, New York, 1972.

Reprinted in Collins (ed.), *Readings in Cognitive Science*, Morgan Kaufmann 1988.

[5] A. M. Collins and M. R. Quillian. Retrieval time from semanticmemory. *Journal of Verbal Learning and Verbal Behavior*, 8:240–248, 1969.

[6] J. H. Condon and Ken Thompson. Belle chess hardware. In M.R.B. Clarke, editor, *Advances in Computer Chess 3*, pages 45–54. Pergamon Press, 1982.

[7] Adriaan D. deGroot. *Thought and Choice in Chess*. Mouton, The Hague, 1965.

[8] Adriaan D. deGroot and Fernand Gobet. *Perception and Memory in Chess*. Van Gorcum, Assen, The Netherlands, 1996.

[9] Chrilly Donninger. CHE: A graphical language for expressing chess knowledge. *ICCA Journal*, 19(4):234–241, 1996.

[10] Charles Ebeling. *All the Right Moves: A VLSI Architecture for Chess*. The ACM Distinguished Dissertation Series. MIT Press, 1987.

[11] Micheal George and Jonathan Schaeffer. Chunking for experience. *ICCA Journal*, 13(3):123–132, 1990.

[12] Jim J. Gillogly. The technology chess program. *Artificial Intelligence*, 3:145–163, 1972.

[13] Fernand Gobet. A computer model of chess memory. In *Proceedings of the 15th Annual Meeting of the Cognitive Science Society*, pages 463–468, 1993.

[14] Fernand Gobet and P. Jansen. Towards a chess program based on a model of human memory. In H. J. van den Herik, I. S. Herschberg, and J. W. H. M. Uiterwijk, editors, *Advances in Computer Chess 7*, pages 35–60. University of Limburg, 1994.

[15] Fernand Gobet and Herbert A. Simon. Templates in chess memory: A mechanism for recalling several boards. *Cognitive Psychology*, 31:1–40, 1996.

[16] J. R. Hayes. Memory organization and world-class performance. In *Proceedings of the 21st Carnegie-Mellon Symposium in Cognition*, 1987.

[17] Dennis H. Holding. *The Psychology of Chess Skill*. Lawrence Erlbaum Associates, 1985.

[18] Feng-Hsiung Hsu. A two-million moves/s CMOS single-chip chess move generator. *IEEE Journal of Solid-State Circuits*, 22(5):841–846, 1987.

[19] Robert M. Hyatt, B. E. Gower, and H. L. Nelson. Cray Blitz. In Don Beal, editor, *Advances in Computer Chess 4*, pages 8–18. Pergamon Press, Oxford, UK, 1985.

[20] Andreas Junghanns and Jonathan Schaeffer. Search versus knowledge in game-playing programs revisited. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, Nagoya, Japan, 1997.

[21] Hermann Kaindl. Positional long-range planning in computer chess. In M. R. B. Clarke, editor, *Advances in Computer Chess 3*, pages 145–167. Pergamon Press, 1982.

[22] Yaakov Kerner. Case-based evaluation in computer chess. In M. Keane, J.P. Haton, and M. Manago, editors, *Topics in Case-Based Reasoning (EWCBR-94)*, Lecture Notes in Artificial Intelligence, Berlin, 1994. Springer-Verlag.

[23] Yaakov Kerner. Learning strategies for explanation patterns: Basic game patterns with application to chess. In M. Veloso and A. Aamodt, editors, *Proceedings of the 1st International Conference on Case-Based Reasoning (ICCBR-95)*, volume 1010 of *Lecture Notes in Artificial Intelligence*, pages 491–500, Berlin, 1995. Springer-Verlag.

[24] Wolfgang Klimesch. *The Structure of Long-Term Memory: A Connectivity Model of Semantic Processing.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.

[25] David N. Levy, editor. *Computer Chess Compendium.* Batsford Ltd., London, 1988.

[26] Donald Michie and Ivan Bratko. Comments to 'chunking for experience'. *ICCA Journal*, 18(1):18, March 1991.

[27] Stephen Muggleton. Inductive acquisition of chess strategies. In J. E. Hayes, D. Michie, and J. Richards, editors, *Machine Intelligence 11*, chapter 17, pages 375–387. Clarendon Press, 1988.

[28] Allen Newell, Cliff Shaw, and Herbert A. Simon. Chess playing programs and the problem of complexity. *IBM Journal of Research and Development*, 2:320–335, October 1958. Reprinted in [25].

[29] John Nunn. *Secrets of Rook Endings.* Batsford, 1992.

[30] John Nunn. Extracting information from endgame databases. In H. J. van den Herik, I. S. Herschberg, and J. W. H. M. Uiterwijk, editors, *Advances in Computer Chess 7*, pages 19–34. University of Limburg, 1994.

[31] John Nunn. *Secrets of Pawnless Endings.* Batsford, 1994.

[32] John Nunn. *Secrets of Minor-Piece Endings.* Batsford, 1995.

[33] Andreas L. Opdahl and Björnar Tessem. Long-term planning in computer chess. In H. J. van den Herik, I. S. Herschberg, and J. W. H. M. Uiterwijk, editors, *Advances in Computer Chess 7.* University of Limburg, 1994.

[34] A. J. Roycroft. Expert against oracle. In J. E. Hayes, D. Michie, and J. Richards, editors, *Machine Intelligence 11*, pages 347–373. Oxford University Press, Oxford, UK, 1988.

[35] Claude E. Shannon. Programming a computer for playing chess. *Philosophical Magazine*, 41(7):256–275, 1950. Reprinted in [25].

[36] Herbert A. Simon and Michael Barenfeld. Information-processing analysis of perceptual processes in problem solving. *Psychological Review*, 76(5):473–483, 1969.

[37] Herbert A. Simon and William G. Chase. Skill in chess. *American Scientist*, 61(4):394–403, 1973.

[38] Herbert A. Simon and Kevin Gilmartin. A simulation of memory for chess positions. *Cognitive Psychology*, 5:29–46, 1973.

[39] David J. Slate and Lawrence R. Atkin. Chess 4.5 — the northwestern university chess program. In *Chess Skill in Man and Machine*, chapter 4, pages 82–118. Springer-Verlag, 2 edition, 1983.

[40] Alan M. Turing. Chess. In B.V. Bowden, editor, *Faster Than Thought*, pages 286–295. Bitman, London, 1953. Reprinted in [25].

[41] David E. Wilkins. Using patterns and plans in chess. *Artificial Intelligence*, 14(3):165–203, 1980.

[42] David E. Wilkins. Using knowledge to control tree search searching. *Artificial Intelligence*, 18(1):1–51, 1982.

[43] Franz-Günter Winkler. Das Vernetzungsmodell von Klimesch - eine Simulation. Master's thesis, Institute for Medical Cybernetics and Artificial Intelligence, University of Vienna, Vienna, Austria, 1991. In German.

[44] Albert L. Zobrist and Frederic R. Carlson. An advice-taking chess computer. *Scientific American*, pages 93–105, June 1973.