SCIENTIFIC AMERICAN October 1990 Volume 263 Number 4

A Grandmaster Chess Machine

In the 40 years since this magazine published the original prospectus for a chess computer, machines have vanquished first novices, then masters and now grandmasters. Will Gary Kasparov be next?

by Feng-hsiung Hsu, Thomas Anantharaman, Murray Campbell and Andreas Nowatzyk

In January of 1988, at a press conference in Paris, world chess champion Gary K. Kasparov was asked whether a computer would be able to defeat a grandmaster before the year 2000. "No way," he replied, "and if any grandmaster has difficulties playing computers, I would be happy to provide my advice."

Ten months after Kasparov's statement, in a major tournament held in Long Beach, Calif., Grandmaster Bent Larsen, a former contender for the world title, was defeated by a chess-playing machine we had designed in a graduate project at Carnegie-Mellon University. The machine, a combination of software and customized hardware called Deep Thought, won five other games, drew one and lost one, tying Grandmaster Anthony Miles for first place. Because machines are disqualified from winning money in tournaments, Miles pocketed the first prize of $10,000. (Deep Thought nonetheless defeated Miles a year later in an exhibition play-off match.)

By the summer of 1990—by which time three of the original Deep Thought team had joined IBM—Deep Thought had achieved a 50 percent score in 10 games played under tournament conditions against grandmasters and an 86 percent score in 14 games against international masters. Some of these games and dozens of others against less distinguished opponents had been played under the auspices of the U.S. Chess Federation, which used the results to derive a chess rating of 2552. That rating indicates a playing strength in the bottom half of the grandmaster range. An average tournament player, by contrast, is rated around 1500 [*see illustration on page 47*]. In the games played after August of 1988, when the computer reached its current analytical speed of 750,000 positions per second, its performance rating exceeded 2600.

The next generation of the machine, expected to play its first game some time in 1992, will run on far more powerful hardware. The equipment will increase the speed of analysis by more than 1,000-fold, to about a billion positions per second. This change alone might well make Deep Thought's descendant a stronger chess player than Kasparov—or any other human being in history.

Why would anyone want to teach a machine how to corner a wooden king on a checkered board? First, chess has long been regarded in the West as the preeminent game of wits and therefore—in Goethe's words—"the touchstone of the intellect." Many people argue that a successful chess machine would prove that thinking can be modeled or, conversely, that chess does not involve thinking. Either conclusion would surely change the conception of what is commonly called intelligence.

Furthermore, computer chess presents an appealing engineering problem. The case was stated in the pages of this magazine 40 years ago by Claude E. Shannon, the founder of information theory [see "A Chess-Playing Machine," by Claude E. Shannon; SCIENTIFIC AMERICAN, February, 1950]:

> The investigation of the chess-playing problem is intended to develop techniques that can be used for more practical applications. The chess machine is an ideal one to start with for several reasons. The problem is sharply defined, both in the allowed operations (the moves of chess) and in the ultimate goal (checkmate). It is neither so simple as to be trivial nor too difficult for satisfactory solu-

FENG-HSIUNG HSU, THOMAS ANANTHARAMAN, MURRAY CAMPBELL and ANDREAS NOWATZYK constructed Deep Thought, the world's leading chess machine, while completing doctorates in various fields of computer science at Carnegie-Mellon University. Hsu, Anantharaman and Campbell have since joined the IBM Thomas J. Watson Research Center; Nowatzyk is with Sun Microsystems. Hsu began the project, worked on it full-time and acted as the system architect. He received a B.S. in electrical engineering from National Taiwan University. Anantharaman wrote most of the host software and implemented various algorithms. He received a B.S. in electrical engineering from Banaras Hindu University in India. Campbell maintained the chess opening book and wrote the precomputation software for the evaluation function. He received B.S. and M.S. degrees in computer science from the University of Alberta. Nowatzyk designed and implemented the automatic tuning of the evaluation function. He received diplomas in physics and computer science from the University of Hamburg in West Germany.

WORLD CHAMPION Gary Kasparov poses with an IBM PS/2, used to communicate with Deep Thought, before beginning a match against the machine late in 1989. Kasparov won despite Deep Thought's grandmaster-strength rating.

tion. And such a machine could be pitted against a human opponent, giving a clear measure of the machine's ability in this type of reasoning.

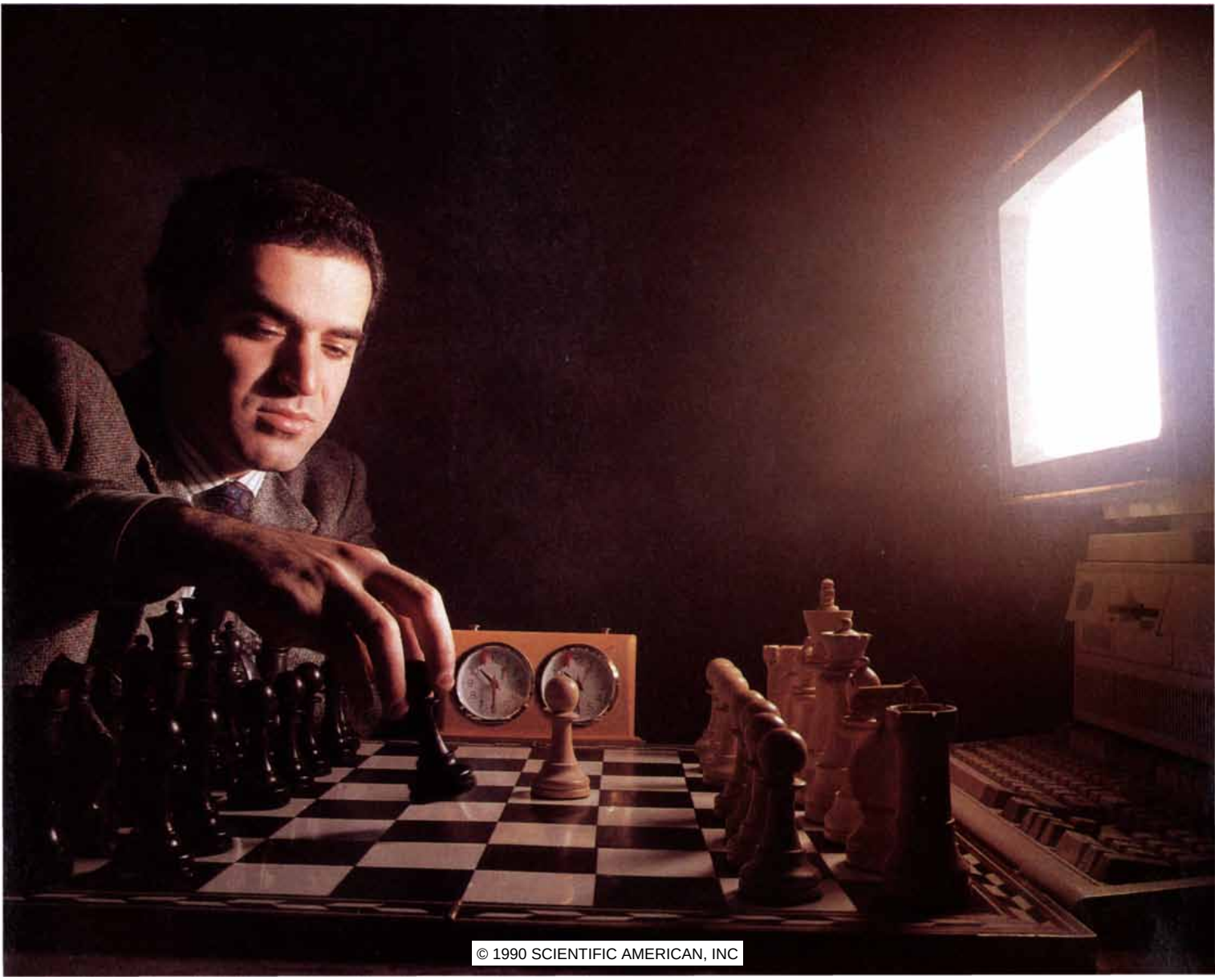
Perhaps the greatest practical consequence of chess programming comes from its demonstration of the efficacy of computer analysis. The perfection of related techniques promises to advance network design, chemical modeling and even linguistic analysis.

The idea of a chess-playing machine dates back to the 1760s, when Baron Wolfgang von Kempelen exhibited the Maezal Chess Automaton in Europe. The machine, nicknamed the Turk because it played its moves by means of a turbaned and mustachioed marionette, was apparently actuated by a complicated mechanism in a cabinet underneath. It generally played well and once sent Napoleon Bonaparte into a fury by beating him in 19 moves. Edgar Allan Poe, among others, later guessed the automaton's secret—a diminuitive chess master made its moves from a secret compartment—but Poe cited the wrong reason: he argued that the Turk's occasional losses were inconsistent with the perfection of a true machine.

Alan M. Turing, the British mathematician, computer scientist and cryptographer, was among the first to consider the problem of a chess-playing computer. He found it easier, however, to work his simple move-generating and position-evaluating program by hand than by machine. Konrad Zuse in Germany and other workers made similar efforts, but the seminal work was done by Shannon. He built on the discoveries of John von Neumann and Oskar Morgenstern, who, in their general theory of games, had devised a so-called minimax algorithm by which the best move can be calculated.

The process basically represents an arbitrarily large number of positions that might result from every possible series of moves, assigns them a numerical score and works backward from this information to derive the best first move. It begins when a move generator calculates all the moves the computer might play from the position at hand, then all the opponent's possible replies and so forth. Each step along the chain of events is called a half move in chess parlance or a ply in the terminology of computer science.

Each new ply in the branching tree of analysis encompasses roughly 38 times as many positions (the number of moves in a typical chess position) as did the previous one, or six times as many positions if "alpha-beta pruning" is used [*see box on page 48*]. Most positions therefore reside in the tree's outermost buds, from which the tree grows until either the game or the computer's allotted time is exhausted. An evaluation function then scores each end position, assigning perhaps "1" to a checkmate of the opponent, "–1" to a checkmate by the opponent and "0" to



a draw. More nuanced advantages can also be registered and balanced against one another. The computer can, for example, count material values—those of pieces and pawns—and calculate positional values according to parameters that represent piece placement, pawn structure, occupation of an unobstructed vertical line of squares (called a file), control of the center and so on.

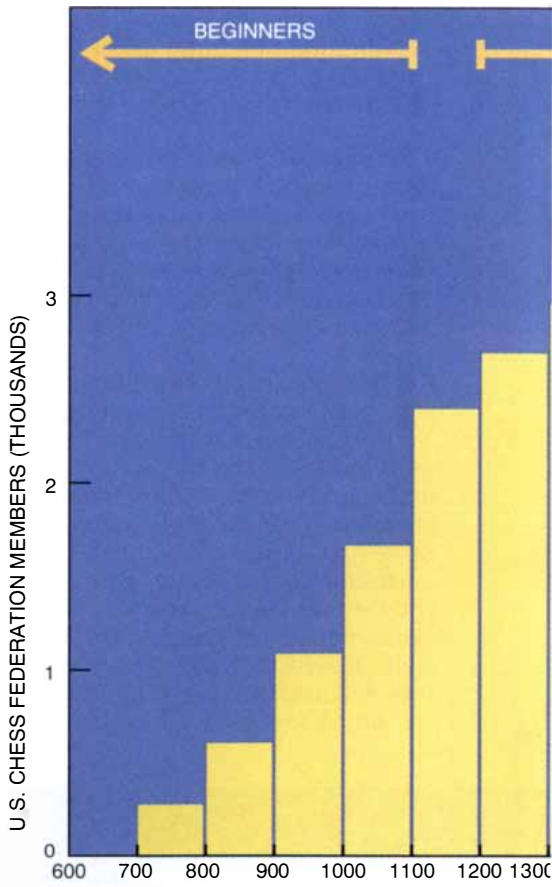
One can strengthen a computer's play by improving its search ability or refining its positional judgment. Flawless play would result if a computer could generate all possible games and classify the end positions as mating or being mated, or neither. Such a computer might surprise its opponent on the first move by announcing, "White to play and mate in 137 moves," or, alternatively, by resigning the position as hopeless. Such exhaustive analysis is easy in games as simple as tic-tac-toe but impractical in chess, in which 10^{120} discrete games are possible. Equally perfect play could be obtained from an examination of only one ply, provided the positional evaluation were as good as that jocularly claimed by Richard Réti, a master who flourished in the 1920s, when he said that he saw only one move ahead—the best.

Such pretensions were far from the minds of the earliest chess programmers, who were not even able to program machines to observe the rules of chess until 1958. Another eight years passed before MacHack-6, a program written by Richard D. Greenblatt of the Massachusetts Institute of Technology, became the first computer to reach the standard of average tournament players.

As the number of people developing chess programs increased, they divided into two philosophical camps. Let us call them the emulation camp and the engineering camp. The former asserted that chess computers should play as humans do, perhaps by means of explicit reasoning about move decisions. The latter took a less restrictive view, arguing that what works well for humans may not be applicable to computers. The emulation camp had most of the say in the early days, when computer chess was more a matter of theory than practice.

In the 1970s the engineering camp moved to center stage when the depth of search was found to correlate almost linearly with the program rating. Each additional ply added about 200 rating points to the computer's playing strength [*see illustration on opposite page*]. Programmers therefore scrambled for access to ever faster computers and found engineering tricks by which to squeeze deeper searches from the available processing power.

MOUNTING CHALLENGE of computers is seen in this superposition of the U.S. Chess Federation's 35,000 rated members and the ratings that are achieved by machines at varying depths of analysis (*green*). Computers gain about 200 points for each additional half move, or ply, they examine. Deep Thought now searches 10 plies for a strength near 2600. Its successor will analyze 14 or 15 plies for a vastly higher rating.



Managing the search is half the battle. At the very beginning of chess programming, search routines generated positions with practically no discernment. They treated transposing variations—which lead to the same position—as if they were distinct. Such needless double counting is now avoided by keeping track of positions in memory arrays known as hash tables. Hash tables provided an even greater benefit by helping the alpha-beta algorithm to weed out many irrelevant lines of play.

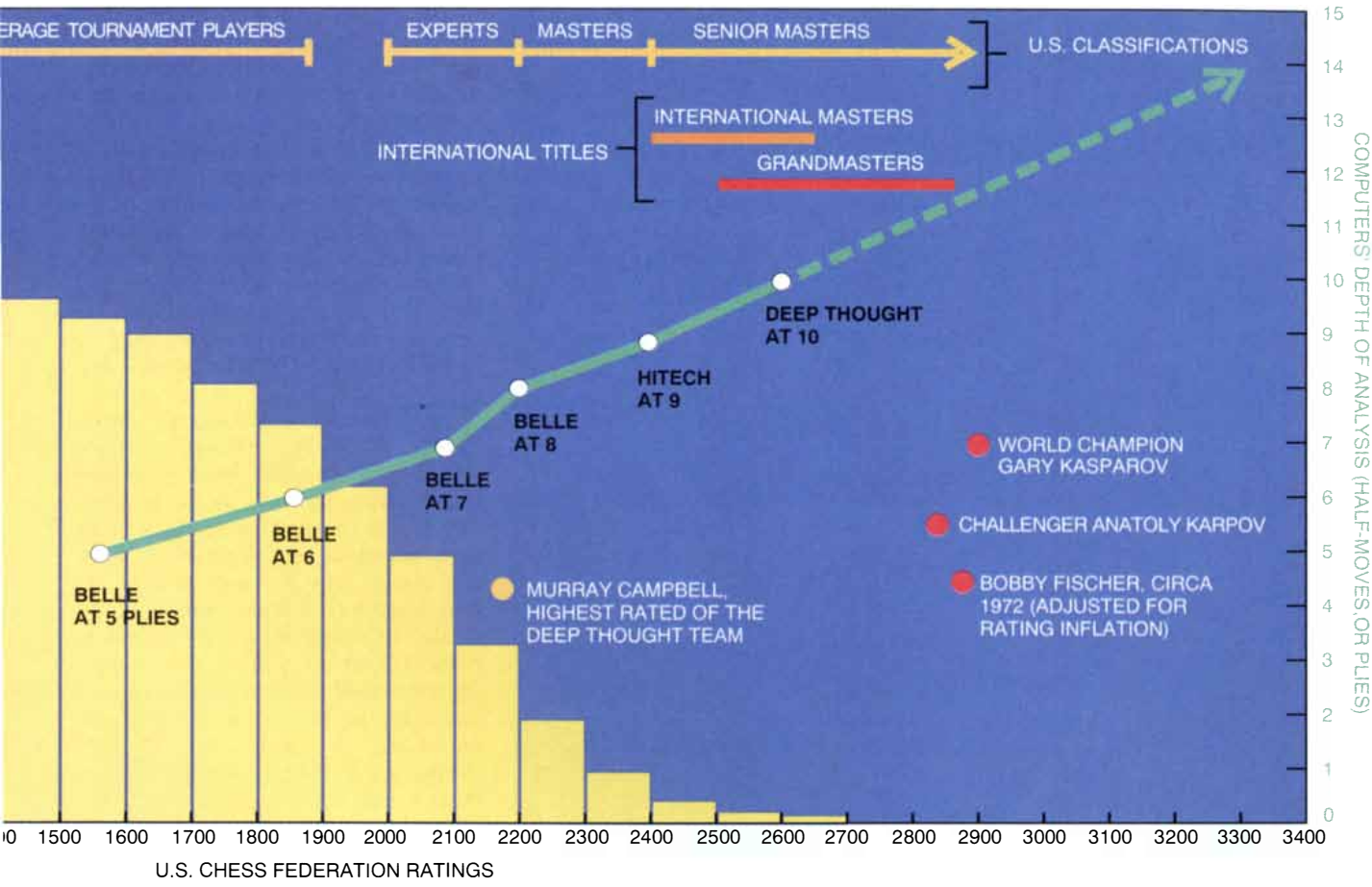
The greatest problem in a search is knowing where to end its multitudinous branches. One cannot examine all lines indefinitely, but one would like at least to avoid cutting off analysis in unstable positions. Such positions result when the analysis stops in the middle of an exchange of pieces. Suppose, for example, the computer searches exactly eight plies ahead in all lines and discovers an eighth-ply position in which it seems to have won a knight in exchange for a pawn. Even if the very next move will enable the opponent to recover the knight and remain a pawn to the good, the computer will steer tenaciously toward the illusory material advantage.

This so-called horizon effect can cause computers to commit a form of chess suicide not seen in the games of even the weakest human players. Out of the blue and for no reason apparent to the naive observer, the machine will begin to throw away its pawns and pieces, leaving its position in tatters. To reduce the chance of such errors, virtually all programs now add a stage of quiescence search to the basic search. Such searches typically examine only sequences of captures of pawns or pieces until reaching a stable, or quiescent, position that is suitable to static evaluation.

The 1970s and early 1980s saw the prevalence of what were commonly called brute-force machines, because they owed their strength to sophisticated implementation of the basic and quiescence search strategies. The period was dominated almost continuously by the Northwestern University program Chess 4.0 and its 4.X descendants. The Northwestern program hopped from one generation of computing hardware to another, steadily increasing its rating until it surpassed expert level (2000) in 1979.

Several attempts to build special-purpose chess machines also began in the 1970s. The most famous one, AT&T Bell Laboratories' Belle, crossed the national master barrier of 2200 in 1983. The era of pure brute-force machines reached its peak around 1986 with the ascendancy of Cray Blitz, which ran on the Cray supercomputers, and Hitech, a special-purpose machine that generated moves on 64 chips—one for each square. Hitech won the 1985 North American Computer Chess Championship, and Cray Blitz won the 1986 World Computer Chess Championship on a tiebreaker by beating Hitech in the last round. Cray Blitz and Hitech searched 100,000 and 120,000 positions per second, respectively.

Deep Thought had a rather unusual history. First, it was developed by a team of graduate students who had no official sponsorship or direct faculty supervision. (Faculty members



conducting work on computer chess at Carnegie-Mellon had no connection with the Deep Thought team.) Second, the team's members had diverse backgrounds that led them to adopt unorthodox approaches.

In June of 1985 one of us (Hsu) concluded that a single-chip move generator could be built with the Very Large Scale Integration (VLSI) technology that was provided to the academic community by the Defense Advanced Research Programs Agency (DARPA). Hsu based his chip on Belle's move generator but found several refinements that made the design amenable to VLSI implementation. He also designed the chip so that its electronic features (including 35,925 transistors) could be packed efficiently, despite the rather coarse (three-micron) minimum-feature size offered by MOSIS, the silicon broker that provided DARPA-funded fabrication services. Hsu spent six months working on design, simulation and layout and then waited four months to receive the first working copies. He tested the chip by linking it to a scientific workstation, and he found that the chip could process up to two million moves per second, 10 times faster than Hitech's 64-chip array.

At this point, Hsu joined forces with Thomas Anantharaman, then a computer science graduate student in the university's speech-recognition group. Anantharaman had written a toy chess program that generated its moves via a software package. By substituting Hsu's chip tester for the package, Anantharaman speeded the program's analysis by 500 percent, to a total of 50,000 positions per second.

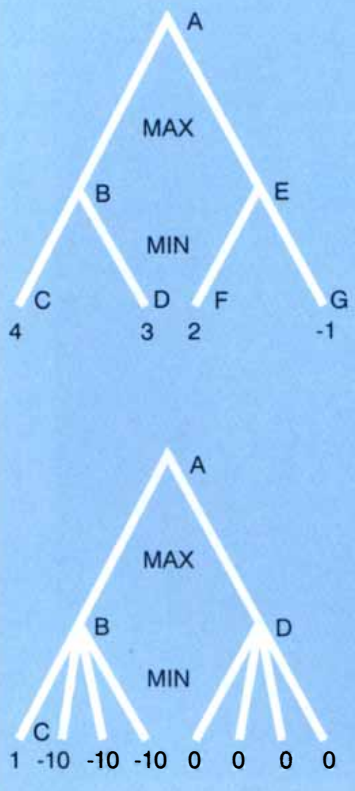
Hsu and Anantharaman became ambitious and, as a lark, decided to prepare their machine for the 1986 North American Computer Chess Championship, then only seven weeks away. Murray Campbell and Andreas Nowatzyk, two graduates in computer science, were then recruited to the project. A more sophisticated evaluation function was desirable, and Campbell, who had once played competitive chess, agreed to work on it. The second and even harder task, given the time constraint, was to augment the chip tester so that it could act as a simple searching engine. Such an engine would exploit the potential speed of the move-generator chip more fully than the workstation linkup had.

Hsu took drastic action to get this component ready in time: he decided the engine would ignore two basic aspects of chess: castling of the king and rook and the repetition of positions. (Any player can claim a draw by demonstrating that a position has been repeated three times, with the same player to move in each case.) To compensate for the omissions, a hybrid search strategy was adopted wherein the early plies were searched on a host computer that considered castling and repetition of positions. The later plies (which of course accounted for most of the positions) were analyzed on the engine.

We had no budget and so created our first machine, ChipTest, by scrounging parts from other projects. The value of all the components did not exceed $500 or $1,000—if the estimated unit cost of the DARPA-funded chips is taken into account. But neither the engine nor the host software was fully debugged in time for the championship, and the defective machine merely managed to obtain an even score. Still, that was not a bad result for seven weeks' work.

We learned much from this debut. Hsu observed, for example, that two other programs had played into lines in which their every move could be forced and where neither side knew the outcome in advance. In other words, the program that emerged with the better position owed its advantage to blind luck. Hsu proposed to remedy the defect with what he called the sin-

TREE-SEARCHING TECHNIQUES



To derive the best first move, a chess program generates a tree of variations, evaluates the final positions and works backward from them. Positional scores are made from the program's point of view; a high score, therefore, is advantageous.

A simple technique, called the minimax algorithm, winnows out the best variations by seeking maximum scores for the computer's moves and minimum scores for the opponent's replies. In the tree on the top, nodes B and E are minimized to 3 and -1, respectively, and node A is maximized to 3.

This simple approach can search an additional ply only by increasing processing power by a factor of 38 (the number of moves in a typical chess position). Alpha-beta pruning refines the algorithm by allowing the program to ignore irrelevant lines of play, so that a sixfold speedup suffices to analyze an extra ply. If, for example, the scoring begins at node C and proceeds to the right, the computer will assign B a value of 3 and then notice that E is less than or equal to 2. It therefore need not examine F.

Another method, called singular extension, gets more out of a computer's search by focusing on critical positions. In the tree on the bottom, the value of B depends heavily on that of C, whereas the value of D does not depend on any one of its successors. To increase the reliability of A's score, the singular extension algorithm would therefore search C one ply deeper than normal. This method allows Deep Thought to see extraordinarily far ahead in many tactically complicated positions.

gular extension algorithm. The algorithm looks deeper, that is, it extends the search to greater depths, in lines where the computer sees only a single good reply. The goal is to ensure that critical positions are given particularly careful attention [*see illustration above*].

When one side is about to win, say, a trapped bishop, the defender usually has fewer and fewer good replies the further the search depth proceeds. Toward the end there is only one good reply, after which the bishop is finally lost. Singular extension homes in on such cases. In one game, it enabled the computer to shock a master by announcing mate within 19 moves.

Anantharaman, the only person who understood the code he had written for the ChipTest host computer, programmed the singular extension algorithm on his own. Meanwhile Hsu completed the microcode, the instructions that control the hardware on its most elementary level. Searching at between 400,000 and 500,000 positions per second, ChipTest won the 1987 North American Computer Chess Championship in a clean sweep, defeating, among others, the world champion machine, Cray Blitz. Thus ended the reign of pure brute force. These days, almost all top programs incorporate at least some elements of selective search.

Our work had made it clear that ChipTest's hardware could be speeded up and that the search could be managed more intelligently. The roughly $5,000 in seed money for this new endeavor—called the Deep Thought project—was provided by Hsu's adviser, H. T. Kung.

The basic version of Deep Thought's chess engine contains 250 chips, including two processors, that plug into a single circuit board measuring about half again the length of this magazine [*see illustration on page 50*]. The engine is managed by a program—the so-called host software—running on a workstation. The machine's processors have hardly more raw speed than do ChipTest's, but improved control of the search algorithm enables them to search 30 percent more efficiently.

The evaluation hardware has four components. A piece placement evaluation (the only inheritance from ChipTest) scores pieces according to their central placement, their mobility and other considerations. A pawn structure evaluation scores pawns according to such parameters as their mutual support, their control of the center of the board and their protection of the king. A passed-pawn evaluation considers pawns that are unopposed by enemy pawns and can therefore be advanced to the eighth rank and promoted to queens. A file structure evaluation assigns values to more complicated configurations of pawns and rooks on a particular file.

We also began to consider ways of tuning the evaluation function's 120 or so parameters, specified in software. Traditionally, programmers had hand-tuned the weights that programs assigned to material—pawns and pieces—and to positional considerations. We believe ours is the only major program to tune its own weights automatically.

We acquired 900 sample master games and arbitrarily defined the optimum weights as those that produce the best match between the moves the machine judges to be best and those that the masters actually played. The software part of the evaluation function was completely rewritten by Campbell and Nowatzyk to reflect this strategy. Instead of just assigning a final numerical value to each position, the evaluation function—in its tuning mode—returns an equation containing a string of linear terms. In other words, it produces a vector.

Two tuning mechanisms were used. The first, which is called hill climbing, simply sets a given evaluation parameter at an arbitrary value and then performs, say, a five- or six-ply search on every position in the game data base to find the moves that the machine would play. It then adjusts the parameter and recalculates. If the number of matches between the computer's choices and the grandmaster's choice should increase, then the parameter is adjusted again in the same direction. The process continues until all the parameters have reached their highest level of performance. It would take years to optimize all the parameters by this method, however, and so we used it only in a few difficult cases.

The second tuning mechanism, proposed and implemented by Nowatzyk, was much quicker. It evolved from the simple notion of finding the best fit between the function of the machine's evaluation of positions and their presumed true values. The best fit provides the lowest average squared value of the error between the model and the true value. True values can be approximated, for these purposes, by the results returned from deep searches (if a known concept is being fine-tuned) or by comparing machine decisions with those of first-rate human players.

The sample games give strong hints about the relative values of positions: any position reached after a grandmaster's move is, after all, likely to

be better than all of the others that would have been reached via alternative moves. Instead of computing a position's value from its parameters, Nowatzyk calculated the parameters on the basis of an assumed difference between the position a grandmaster chose and the alternative positions the grandmaster had rejected. His algorithm takes only a few days to compute, and unlike hill climbing, it does not improve parameters one at a time but improves the entire set of parameters simultaneously.

Our automatically tuned evaluation function appears to be no worse, if no better, than the hand-tuned functions of such well-known academic chess programs as Hitech and Cray Blitz. There still appears to be a gap, however, between Deep Thought's evalua-

ANATOLY KARPOV VERSUS DEEP THOUGHT

HARVARD UNIVERSITY, FEBRUARY 1990

Comments by International Master Michael Valvo

WHITE KARPOV	BLACK DEEP THOUGHT
1 e2-e4	c7-c6
2 d2-d4	d7-d5
3 Nb1-d2	g7-g6
4 c2-c3	Bf8-g7
5 e4-e5	

This move marked the end of Deep Thought's opening book.

...	f7-f6!

Fantastic play by a machine! Black immediately attacks the head of the pawn chain and induces White to weaken his control of the light squares.

6 f2-f4	Ng8-h6
7 Ng1-f3	castles
8 Bf1-e2	f6xe5!

Deep Thought finds active counterplay.

9 f4xe5	c6-c5!

So that 10 d4xc5 would allow Black to obtain some advantage with Nf3-g4!

10 Nd2-b3	c5xd4
11 c3xd4	Nb8-c6
12 castles	Qd8-b6
13 Kg1-h1	a7-a5

White's central pawn phalanx still guarantees him an edge, but his pawn on d4 is a major weakness and his pieces neither cooperate well nor occupy good squares.

14 a2-a4	Bc8-f5
15 Bc1-g5	Bf5-e4
16 Nb3-c5!	

This move, which threatens a simultaneous attack on Black's Queen and Rook on the next move, is probably based on a little insight into the way computers operate. Deep Thought cannot resist capturing the pawn:

...	Qb6xb2?

A mistake; better was 16 ... Nh6-f5!, preparing to sacrifice a Rook for a Knight after 17 Nc5-d7, Qb6xb2; 18 Nd7xf8, Nf5xd4! when Black's attack would have been very dangerous.

17 Nc5xe4	d5xe4

Deep Thought now thinks that it is down by the equivalent of one third of a pawn. Karpov later was impressed with the computer's evaluation, which approximated his own during the game.

18 Ra1-b1	Qb2-a3

Forced, as 18 ... Qb2-c3 loses to 19 Rb1-b3, whereas 18 ... Qb2-a2 runs into 19 Nf3-d2 followed by 20 Be2-c4.

19 Bg5-c1	Qa3-c3
20 Bc1-d2	Qc3-a3
21 Bd2-c1	Qa3-c3
22 Rb1-b3	Qc3-a1

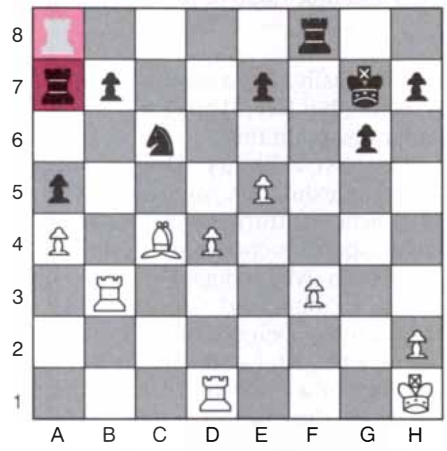
Karpov repeats moves to gain thinking time.

23 Be2-c4(check)	Kg8-h8
24 Bc1xh6!	Qa1xd1
25 Bh6xg7(ch)	Kh8xg7
26 Rf1xd1	e4xf3
27 g2xf3	

27 Rb3xb7 was better. But how could Karpov have guessed Deep Thought's next move?

...	Ra8-a7!!

DEEP THOUGHT



KARPOV

The audience laughed at this "ugly" move. According to Karpov, however, it is Deep Thought's only chance.

28 Bc4-d5	Rf8-d8
29 Rb3-b5	Ra7-a6!

The computer defends very resourcefully. Now it threatens 30 ... Nc6-a7; 31 Bd5xb7, Na7xb5; 32 Bb7xa6, Rd8xd4; with equality.

30 Bd5-c4	Ra6-a7
31 Bc4-d5	Ra7-a6
32 Rb5-c5	Rd8-d7
33 Kh1-g2	Ra6-b6!
34 Bd5xc6	b7xc6
35 Kg2-f2!	

Risky, given Black's slightly superior game, but Karpov still wants to win.

...	Rd7-d5
36 Rc5xd5	c6xd5
37 Rd1-c1	Rb6-b4
38 Kf2-e3	Rb4xa4

Another, and perhaps simpler, way to draw is 38 ... Rb4-b3(ch); 39 Ke3-e2, Rb3-b4 with repetition of positions, as White can hardly afford to give up his d-pawn.

39 Rc1-c5	e7-e6
40 Rc5-c7(ch)	Kg7-g8
41 Rc7-e7	Ra4-a3(ch)
42 Ke3-f4	Ra3-d3
43 Re7xe6	Rd3xd4(ch)
44 Kf4-g5	Kg8-f7!

A fine defensive interpolation.

45 Re6-a6	a5-a4

Deep Thought thinks it stands better and therefore refuses to force a draw by 45 ... h7-h6(ch); 46 Kg5xh6, Rd4-h4(ch); 47 Kh6-g5, Rh4-h5(ch); 48 Kg5-f4, Rh5-f5(ch); followed by 49 ... Rf5xe5.

46 f3-f4	h7-h6(ch)
47 Kg5-g4	Rd4-c4?

47 ... g6-g5! draws, but the machine, still thinking it is ahead, refuses to give up a pawn for safety.

48 h2-h4	Rc4-d4
49 Ra6-f6(ch)	Kf7-g7
50 Rf6-a6	Kg7-f7
51 h4-h5	g6xh5?

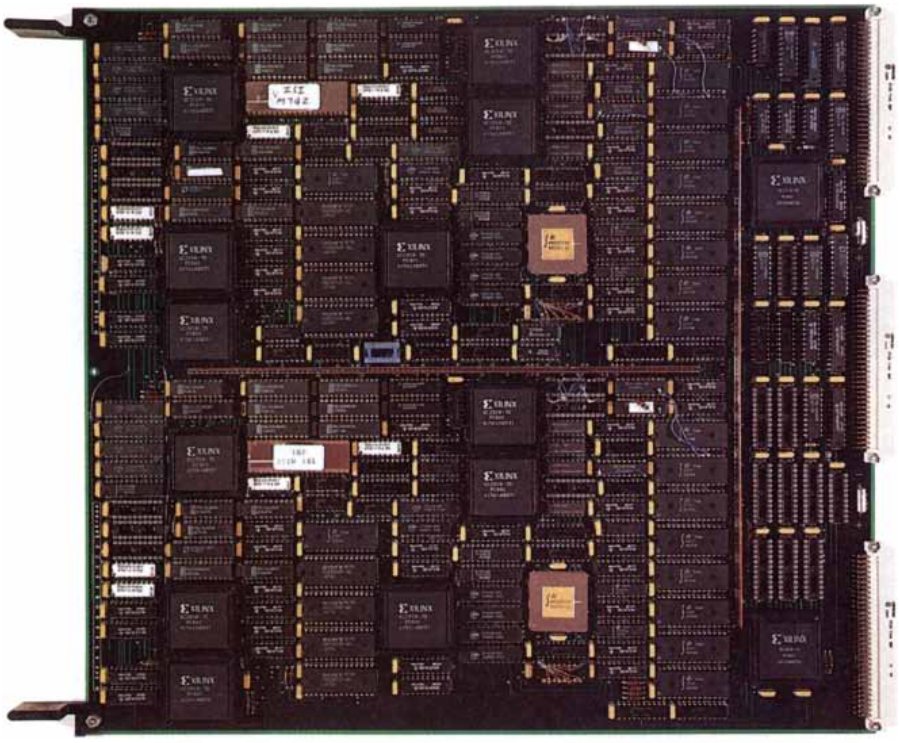
This loses. 51 ... g6-g5 would have held White to an edge too slight to win the game, Karpov later concluded.

52 Kg4-f5	kf7-g7
53 Ra6-a7(ch)	Kg7-f8
54 e5-e6	Rd4-e4
55 Ra7-d7	Re4-c4
56 Rd7xd5	h5-h4
57 Rd5-d3	Kf8-e7
58 Rd3-d7(ch)	Ke7-f8
59 Rd7-h7	h6-h5
60 Kf5-e5	h4-h3
61 f4-f5	Kf8-g8
62 Rh7xh5	a4-a3
63 Rh5xh3	a3-a2

A human player would play 63 ... Rc4-a4, to prolong the struggle, but then 64 Ke5-f6 would soon prevail.

64 Rh3-a3	Rc4-c5(ch)
65 Ke5-f6	Black resigns

Deep Thought considered itself to be down by the equivalent of at least six pawns, so at this point its designers decided to resign for it. The machine had about 20 minutes in which to make its moves, while Karpov had less than a minute left on his clock; however, that was more than enough time for a player of his caliber to force the win.



HARDWARE HEART of Deep Thought fits on a circuit board the size of a large pizza. Each of its two processors can search 500,000 positions per second. The successor machine will shrink Deep Thought to a chip and link 1,000 of them in parallel.

tion function and those of the top commercial chess machines, which typically are the fruit of many man-years of work. With better feedback from the automatic tuning procedures, we hope to close the breach soon.

It may seem strange that our machine can incorporate relatively little knowledge of chess and yet outplay excellent human players. Yet one must remember that the computer does not mimic human thought—it reaches the same ends by different means. Deep Thought sees far but notices little, remembers everything but learns nothing, neither erring egregiously nor rising above its normal strength. Even so, it sometimes produces insights that are overlooked by even top grandmasters.

The machine's inhuman insights were perhaps the reason behind Grandmaster Kevin Spraggett's decision to engage the machine as an assistant in preparing for a World Championship Candidate Quarter Final match with Grandmaster Artur Yusupov. The machine's participation had no discernible effect on the match, but it did establish an interesting precedent.

In October of 1989 an experimental six-processor version of Deep Thought played a two-game exhibition match against Kasparov in New York City. Although the new version was capable of searching more than two million positions per second, Kasparov disposed of it quite easily. The result was not unexpected, but Deep Thought's play was rather disappointing.

This past February, Deep Thought played an exhibition game against Anatoly Karpov, a former world champion and Kasparov's challenger in the 1990 title match (which begins in New York City in October and concludes in Lyons, France). Defects that surfaced in the experimental software in the six- and four-processor versions led us to revert to the two-processor version. Deep Thought, benefiting from a number of improvements in its evaluation function, played one of its best games in the first 50 moves, then blundered away a clearly drawn position. A stable six-processor version would have had enough speed to avoid the blunder [*see illustration on preceding page*].

Speed is the key to work now under way at the IBM Thomas J. Watson Research Center, where the next-generation machine is now being designed. It should outcalculate its predecessor by a factor of at least 1,000. The machine we have in mind will therefore examine more than a billion positions per second, enough to search 14 or 15 plies deep in most cases and from 30 to 60 plies in forcing lines. If the observed relation between processing speed and playing strength holds, the next-generation machine will play at a 3400 level, about 800 points above today's Deep Thought and 500 points above Kasparov's rating record.

To achieve this speed, Hsu is designing a chess-specific processor chip that is projected to search at least three million moves per second—more than three times faster than the current Deep Thought. He is also designing a highly parallel computing system that will combine the power of 1,000 such chips, for a further gain of at least 300-fold. Anantharaman and Campbell are improving various aspects of the current version of Deep Thought, so that these improvements can be incorporated into the next machine as well. Nowatzyk is pursuing other interests.

We believe the system will be strong enough, by virtue of its speed alone, to mount a serious challenge to the world champion. We further believe that the addition of a long list of other planned improvements will enable the machine to prevail, perhaps as soon as 1992.

Kasparov begs to differ, and we respect his opinion. In a private communication, he acknowledged that a machine searching a billion positions per second might defeat the general run of grandmaster, then added, "But not Karpov and me!" Kasparov contended that the very best players should be able to prepare themselves to exploit the special weaknesses presented by machines. He maintained that human creativity and imagination, in particular *his* creativity and imagination, must surely triumph over mere silicon and wire.

When the two opinions collide over the board, the ingenuity of one supremely talented individual will be pitted against the work of generations of mathematicians, computer scientists and engineers. We believe the result will not reveal whether machines can think but rather whether collective human effort can outshine the best achievements of the ablest human beings.

FURTHER READING

BELLE. J. H. Condon and Ken Thompson in *Chess Skill in Man and Machine*. Second Edition. Edited by P. W. Frey. Springer-Verlag, 1984.

CHESS 4.5—THE NORTHWESTERN UNIVERSITY CHESS PROGRAM. David J. Slate and Lawrence R. Atkin in *Chess Skill in Man and Machine*. Second Edition. Edited by P. W. Frey. Springer-Verlag, 1984.

LARGE SCALE PARALLELIZATION OF ALPHA-BETA SEARCH: AN ALGORITHMIC AND ARCHITECTURAL STUDY WITH COMPUTER CHESS. Feng-hsiung Hsu. Ph.D. Thesis. Carnegie-Mellon University Computer Science Department, CMU-CS-90-108, February, 1990.