Computer Chess: Mind vs. Machine

Human chess masters still easily outperform computers. The contrast in their strategies illuminates differences between natural and artificial intelligence.

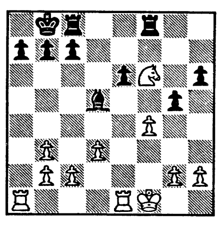
BY LYNN ARTHUR STEEN

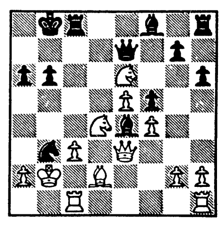
Ever since computers first made their presence felt in human affairs, mankind has been haunted by the unknown specter of an artificial, electronic intelligence usurping man's role as the repository of rational thought on this planet. Early research into the nature of this artificial intelligence often centered on games of strategy because they offered an excellent microcosmic simulation of more realistic "real-life" problems. Simple games like tic-tac-toe posed no problem to the new electronic machines, although the computer's capacity to play perfect tic-tac-toe—sometimes with judiciously selected "errors" inserted to maintain interest—is still a source of amusement among the general population. More complex games, however, posed a significant challenge, both to those trying to understand how to make machines exhibit intelligent behavior and to those trying to understand the nature of man's intelligence. The preeminent example of a game that challenges both artificial and natural intelligence is chess.

Research into computer chess began a quarter century ago when information theorist Claude Shannon proposed a general method for constructing a computer program to play legal and respectable chess. Research efforts since that time have led to about 40 moderately good programs around the world—about 25 of them in the United States alone. At the recent meeting of the Association of Computing Machinery in Minneapolis, Minn., in October, 12 of the best North American programs played each other in a tournament. And all 12 played a simultaneous exhibition match against chess master David Levy of Scotland. Despite fatigue from jet-lag, Levy won 10 and drew 2 of the 12 exhibition games.

So computers still have a long way to go. Levy is looking forward to collecting in three years on a £1,000 bet he made in 1968 that no computer program could beat him in a 10-game match in 10 years. Several of those closest to the computer chess effort predict that it might take as much as a half a century more before a machine's artificial intelligence can match chess wits evenly against the best natural intelligence.

Early pioneers in artificial intelligence—Claude Shannon, Alan Turing, Norbert Weiner—championed chess as an ideal proving ground for research because it posed a problem of sufficient definitiveness to permit assessment of the quality of the work being done, yet of sufficient complexity to prohibit a trivial solution based on the computational ability of the computer. Chess programs require heuristics—rules of thumb—that simulate the process of human thought; thus, chess programs form an excellent counterpoint





in the study of human intelligence.

Shannon estimated that there are more than 10^{120} different possible sequences of moves from the beginning of a standard chess game. The fastest present computers would take more than 10^{90} years to examine all these possibilities; even the fastest theoretically possible computer would take only slightly less time—perhaps 10^{80} years! So whereas computers can "crunch" tic-tac-toe, and even checkers, by looking ahead all the way to the end of the game, they cannot do this with chess. That is why chess is such a challenge for players and for researchers in artificial intelligence.

Shannon's general strategy makes effective chess programs possible even though it is impossible to look very far ahead. His idea—first published in 1950—was based on the then-new theory of John von Neumann and Oskar Morgenstern concerning the "minimax" strategy of game theory for economic analysis. Shannon simply applied von Neumann and Morgenstern's ideas to the game of chess.

First, a program must generate (or grow) a tree of possible moves, responses, and subsequent moves—for as many layers, or ply, as time permits. (Most present programs can manage at most 5 to 6 ply, involving several hundred thousand branches in the tree.) Then the program must evaluate the strength of the board position at each terminal point in this tree, taking into account such generally important factors as the balance of material between the players, the position of the pieces and their mobility in anticipation of future moves. The result is generally thousands, sometimes hundreds of thousands, of potential future board positions with corresponding evaluations.

At this stage the minimax strategy from the theory of games is used to sort out these myriad possibilities. The computer is programmed to assume that the human opponent is smart enough to always select moves that will minimize the value of the resulting board position to the computer. So, in anticipation of this type of "rational" behavior by the human opponent, the computer adopts the rule of choosing—at each branch in the game tree—that one move that will maximize the minimum value that its opponent will attempt to realize. In short, the computer picks that move in which the human opponent can hurt it least.

This strategy leads to conservative play because the computer will pass up a good move if there is a stronger (though perhaps well hidden) rebuttal that the opponent could make. Computer strategy based on the theory of games minimizes risks rather than maximizes opportunities.

Shannon elaborated on this strategy in his seminal paper by suggesting two variations on the central theme: Type A programs would attempt to generate and examine a complete tree of all possible moves to some predetermined depth—based on the physical limits of computer speed and memory. Type B programs would introduce various devices to prune the tree as it grows, so that promising moves can be investigated to a deeper level than less promising ones.

Type A programs are severely limited by the rapid growth in the size of a complete move tree: It would take several hours to perform a complete examination on a six-ply tree. Yet many crucial chess maneuvers require a sequence of moves and captures that extend beyond six moves. Type B programs, on the the other hand, are limited in the sense that they may overlook "sleeper" moves—those that appear innocuous when examined to a few ply, but will turn out to be significant if pushed much deeper. Moreover, the quality of a Type B program will depend not only on the quality of its evaluation routine, but also on the quality of the selection routine that decides which moves warrant deep study and which do not.

Effective programs amend the basic

Continued on page 350

. . . Chess

Shannon strategy in several ways in order to conserve time and improve efficiency. Standard sequences of opening moves are stored in what is called the "book"—a dictionary of board positions commonly encountered in the early moves of a game. Some of these books have as many as 25,000 to 50,000 stored responses, precipitating a rapid-fire opening to computer vs. computer games.

In the middle game, various devices are used to adjust the dual controls of depth and breadth of search; special strategies called alpha-beta algorithms are used to prevent the computer from growing irrelevant branches in the minimax tree. And the better programs employ a variety of end-game routines to orchestrate the few remaining pieces into an effective attack or defense. (Weak programs typically have surprising difficulty pursuing an opportunity to checkmate to its successful conclusion.)

The contrast between machine and human strategy in chess is instructive. Beginning human players tend to generate large move trees, and then pick the best one by means of an informal minimax estimate based on material strength as the prime evaluation—with imminent attack of major pieces as a close second. But no human player ever considers all possible moves; even weak or beginning players will concentrate attention on certain pieces, either because of their value, or because of their position on the board. Very good players—masters and grand masters—will often indicate, if asked to provide a verbal protocol of their moves, that they really only considered two or three moves. In preparing for tournament competition, many chess buffs practice by playing rapid games, allowing only 5 to 10 seconds per move.

Clearly, human players employ an effective device for pruning the game tree, and better players tend to prune the tree more severely than weaker ones. The rules of thumb that enable chess masters to hone in on those few moves that deserve serious consideration, form the heuristics for the artificial intelligence in machine programs. And it takes a chess master to instruct the computer program in appropriate heuristics. None of the programs yet devised is capable of learning or inferring heuristics based on experience.

Nearly all present chess programs follow the Shannon strategy, most emphasizing Type B look-ahead (variable depth of search) with a few emphasizing Type A (comprehensive search). The winning program in the most recent tournament—a program from Northwestern University that has captured the title five out of six times—claims to emphasize both. In other words, the route to success at the moment involves as deep a complete search as possible, followed by selective search to even deeper levels. When the Northwestern program played Levy to a draw in the simultaneous exhibition, it examined about 200,000 board positions for every move!

But this strategy, even though winning tournaments at the moment, is the subject of a good deal of controversy among those working in artificial intelligence. Comprehensive search emphasizes brute force power—that is, massive computation—without the economy of effort that some believe is the mark of real intelligence. David Levy, for instance, cited an excellent computer chess game played by a German program in a recent tournament as evidence that massive computation may not be the only way to mimic intelligent behavior. This program won its game in an impressive series of directed attacks, even though it had no look-ahead features at all.

Al Zobrist of the Jet Propulsion Laboratory in California has worked for several years at the University of Southern California on a totally different program which uses perception of form rather than computation of forthcoming moves as the basic concept: His program perceives and evaluates chess patterns, rather than board positions. Zobrist claims that pattern recognition is a more accurate representation of human intelligence. He cites, for example, the general perception of a need to move a king from one side of the board to another. A human need only recognize that simple thought to know how to carry it out; a computer would need to grow a move tree nearly 20-ply deep to foresee the value of this type of move.

Programs based on pattern recognition (coupled with selected look-ahead to forthcoming patterns) require greater input of "smart heuristics" because the type of patterns that are germane to master-level chess are known only—if at all—by chess masters themselves. Such programs are more responsive to corrective suggestions made by chess masters who observe and diagnose problems; it is far easier to add a new pattern than to amend a minimax algorithm. And they appear in many ways to simulate human chess-play more faithfully than do the Shannon minimax algorithms. Unfortunately, they do not yet play chess as well as the best of the traditional programs.

But not even pattern-recognition programs can simulate human play as well as a mechanical chess machine known as the Turk, introduced in Austria by Baron Wolfgang von Kempeler in 1769. This marvelous automaton toured European and American cities for several decades, conquering chess amateurs wherever it went. It's *élan vital* was neither minimax algorithms nor smart heuristics, but an intricate series of levers manipulated by a chess master concealed within the machine itself. Modern chess machines, if nothing else, are more humane: Instead of capturing the chess master, they capture only his heuristics. ☐