A World-Championship-Level Othello Program

Paul S. Rosenbloom

August 1981

DEPARTMENT
of
COMPUTER SCIENCE

Carnegie-Mellon University

(14) CMU-CS-81-137

(6) # A World-Championship-Level Othello Program,

(10) Paul S. Rosenbloom
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

(11) August 1981

(12) 53

Abstract

Othello is a recent addition to the collection of games that have been examined within artificial intelligence. Advances have been rapid, yielding programs that have reached the level of world-championship play. This article describes the current champion Othello program, ***Iago***. The work described ~~here~~ includes: (1) a task analysis of Othello; (2) the implemenation of a program based on this analysis and state-of-the-art AI game-playing techniques; and (3) an evaluation of the program's performance through games played against other programs and comparisons with expert human play.

(15)

This research was sponsored by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 3597, monitored by the Air Force Avionics Laboratory Under Contract F33615-78-C-1551,

vv ARPA Order - 3597

A

403 081

mt

Table of Contents

List of Figures

List of Tables

1. Introduction

The game of Othello[1] is a modern variation of the 19th century board game Reversi. It was developed in its current form in 1974 by Goro Hasegawa (see Personal Computing (1980) for more details), and up until recently it has been played primarily in Japan, where it is the second most popular board game (next to Go). Othello is similar to backgammon in its appeal; the rules are simple, an enjoyable game can be played by beginners, and there is a great deal of complexity that must be dealt with in order to play the game well. One of the purposes of the present article is to face this complexity by treating the game of Othello as a subject for scientific analysis. The primary purpose of this article is to show how state-of-the-art game-playing techniques can be applied to Othello to yield a competent performance program, called ***Iago***. In its most recent competition, ***Iago*** won the Santa Cruz Open Machine Othello Tournament[2], with an 8 - 0 record against an international field of computer Othello programs. In his review of the tournament, Jonathan Cerf, the current world Othello champion, stated [Cerf 81b]: "In my opinion the top programs from Santa Cruz are now equal (if not superior) to the best human players."

The study of game playing is one of the oldest and most developed portions of artificial intelligence. Early work on checkers [Samuel 63] showed the potential of AI techniques in game playing. More recently, the field has been driven primarily by advances in chess [Slate & Atkin 77], which have led to programs that are fast approaching the master level of play. Work on other games has continued to broaden the range of application of game-playing methods. For example, the work by Berliner on backgammon has produced a program that won a match against the world backgammon champion, as well as contributing new methods and theory to the field [Berliner 80].

The success of ***Iago*** is additional evidence of the power of AI game-playing techniques. When I began this effort, my knowledge of both Othello and game-playing techniques was rudimentary. My total experience with Othello consisted of about ten games against a very poor program (which played the *maximum disc strategy* (Section 3.1.1)). Since then, with about 5 man-months of effort, ***Iago*** has been brought up to world-championship level.

Iago routinely beats all human players around the CMU computer science department (including the author). Because programs do not currently participate in human Othello tournaments, ***Iago*** has only competed in two tournaments organized explicitly for Othello programs. The first -- the Northwestern Man-Machine Othello Tournament -- took place in June 1980. There were eight invited competitors, including the

[1]Othello is CBS Inc.'s registered trademark for its strategy disc game and equipment. Gameboard design (c) 1974 CBS Inc.

[2]A description of the tournament and the final standings can be found in Frey (1981a).

then world Othello champion, Hiroshi Inoue, the U.S. Othello champion, Jonathan Cerf, and an early version of ***Iago***. That version of ***Iago*** was written in the ***Sail*** language, and was running on CMU-10D (a DecSystem 1050 -- KA10 processor with 240K words of primary memory). The other five competitors were Othello programs. ***Iago*** came in fifth -- losing to both human entries, and ending up 3-1-1 versus the other programs. It is noteworthy that each of the humans lost to a program in this tournament.

The second tournament -- the Santa Cruz Open Machine Othello Tournament -- was held in January 1981. This tournament was open to any computer Othello program, and had no human contestants. It drew an international field of twenty programs, including the top entries from the previous tournament. The tournament was eight rounds, with the pairings being made so that programs with comparable records were paired (with duplications eliminated). An improved version of ***Iago*** (still written in ***Sail***, but now running on CMU-20C, a DecSystem 2060 -- KL10 processor with 512K words of primary memory) earned a perfect 8-0 record, including wins against the top four challengers (and seven out of the top nine challengers).

It is difficult to compare the structure of ***Iago*** with that of the other leading Othello programs, because there is a reluctance on the part of most of the authors to release information[3]. This stems from a desire to maintain a competitive edge (and for many, a marketing edge as well). However, one thing is clear -- computing power has not been the primary determinant of success in computer Othello. The top two programs in the Northwestern tournament, and the second through seventh finishers in the Santa Cruz tournament were running on microcomputers. It is the structure and knowledge contained in the current version of ***Iago*** that allow it to perform at a world-championship level.

The topics covered in this paper include a description of the game of Othello and its rules (Section 2); an analysis of Othello (Section 3); the structure and mechanisms in ***Iago*** (Section 4); and a comparison of ***Iago***'s play with expert human play (Section 5). Section 6 contains some concluding remarks.

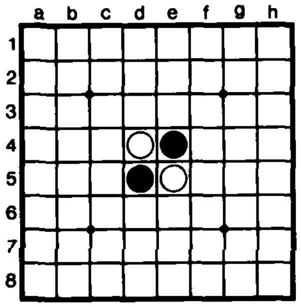
---

[3]For what has been written, see Frey (1980a, 1980b), Maggs (1979), and Phillips (1980). A brief description of ***Aldaron*** (written by Charles Heath), the second place program in the Santa Cruz Open Computer Othello Tournament, appears in Cerf (1981b) (along with a brief description of ***Iago***). To the level at which it is described, ***Aldaron*** appears to employ information not too different from that in ***Iago***.

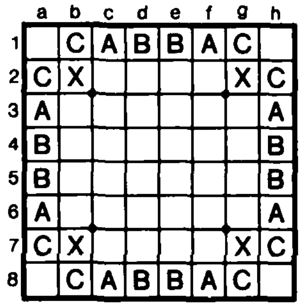
2. The Game of Othello

Conceptually Othello is a derivative of the Go family of board games; emphasizing the capture of territory through the process of surrounding the opponent's pieces. It is played on an 8x8 board, with a set of dual-colored discs. Each disc is black on one side and white on the other. The initial board configuration is shown in Figure 2-1a. Othello notation differs from standard Chess notation in numbering the rows from top to bottom, so the top left-hand corner square is *a1*. In addition, many of the squares have standard names (Figure 2-1b), such as square *b2* which is an *x-square*[4].

Initially, white owns the two central squares on the main diagonal (*d4* and *e5*), and black owns the two central squares on the minor diagonal (*e4* and *d5*). Black plays first, and then the players take turns moving until neither side has a legal move. The player with the most discs at this point is declared the winner (there may be ties). In Othello, a player controls only those squares that have a disc with his color on them.



(a) The initial board configuration



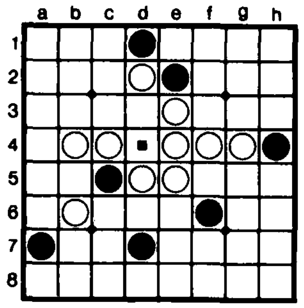
(b) The named board squares

Figure 2-1: The Othello board

A move is made by placing a disc on the board, with the player's color facing up. In order for a move to be *legal*, the square must be empty prior to play, and placing the disc must capture some of the opponent's discs by flipping them to the player's color. Figure 2-2 shows an example of a legal move by black to square *d4*, and the resulting changes in disc ownership. The board position in Figure 2-2a could not possibly have arisen during a game; it is intended solely to illustrate a legal move.

The opponent's discs are flipped by *bracketing* them between the disc being played and an existing disc belonging to the player. Bracketing can occur in a straight line in any of the eight directions (two vertical, two horizontal, and two in each diagonal direction), and consists of an arrangement of the form: the empty square,

[4]See Sullivan and Richards (1981) for a glossary of this and other Othello terms.


(a) Before black's move at *d4*

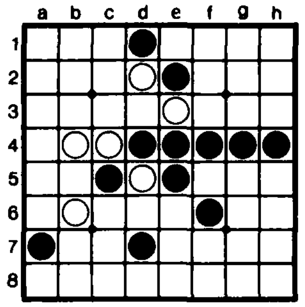

(b) After black's move at *d4*

Figure 2-2: Contrived example to show results of a legal play.
The marker (the solid black square) in (a) points to where the disc will be played.

followed immediately by one or more of the opponent's discs, followed immediately by one of the player's discs. This arrangement can be found in two directions from *d4*: horizontally to the right (flipping the discs at *e4*, *f4*, and *g4*); and on the lower right diagonal (flipping the disc at *e5*). There can be no empty squares within this string, so the disc at *d5* does not flip. *Discovered* bracketing does not cause discs to be flipped; even though the disc at square *e4* is flipped, causing the disc at *e3* to be bracketed, the disc at *e3* is not flipped. All of the opponent's discs that are bracketed by the new disc are flipped to the player's color. No discs are ever taken off the board -- they just change color.

If the player does not have a legal move, he loses his turn and the opponent plays again. Usually the game ends only after the entire board has been filled with pieces, which normally takes 64 - 4 = 60 moves. However, games can terminate before this occurs when neither side is able to move. In the Santa Cruz tournament, seven games (out of the 77 games for which a record is available [Frey 81b], excluding all types of forfeits) ended before the board was filled. Two games ended because of *wipe-outs*, 56-0 and 42-0, and the other five ended with an average of 1.4 empty squares. In tournaments, games can also end when a player forfeits because of lack of time. Each side is allowed 30 minutes in which to make all of their moves. This yields an average of one minute per move, assuming that each side will make half of the moves.

The average winning margin at the Santa Cruz Open was 27.9 discs (standard deviation of 15.7). Figure 2-3 shows how the margin was distributed. Of the 77 games, black won 36 times (average margin of 30.6 discs), and white won 41 times (average margin of 25.5 discs).

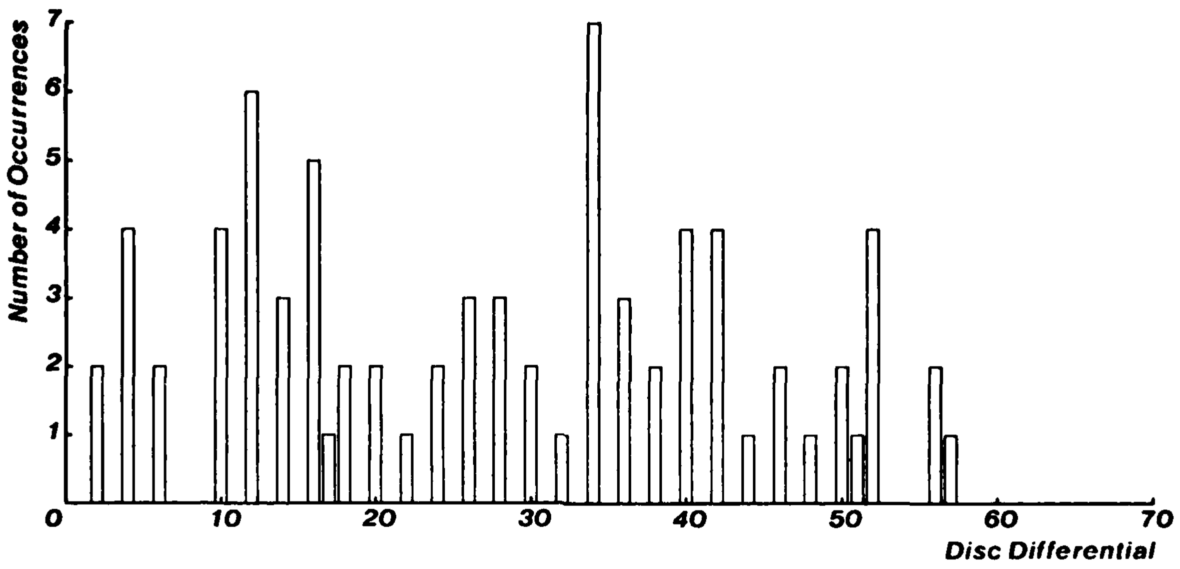


Figure 2-3: Distribution of winning margin over 77 machine games
Santa Cruz Open Machine Othello Tournament

3. An Analysis of Othello

The basis for a program is always a task analysis. Due to its newness, Othello strategy is still in an embryonic stage of development, and useful descriptions of the game are lacking. There is one English-language book on Othello [Hasegawa & Brady 77], but its advice is more misleading than helpful. The analysis in this section is limited to what the author and coworkers (primarily Bruce Ladendorf and Charles Leiserson) have been able to ascertain about the game[5].

3.1. Three simple strategies

A number of simple strategies, such as the obvious one of maximizing the numerical disc advantage, have been suggested for Othello. While none of these strategies can produce a competent performance program, their examination does lead to concepts that are useful in a more detailed task analysis. In this section we examine three single-concept strategies: (1) the obvious one, called the *maximum disc* strategy; (2) the *weighted square* strategy; and (3) the *minimum disc* strategy.

3.1.1. The maximum disc strategy

The maximum disc strategy derives from a straightforward attempt to use the top-level goal of the game (possess the most discs at the end of the game) as a playing heuristic. As such, it is a form of *greedy* algorithm. The goal gets translated into a heuristic of the following form.

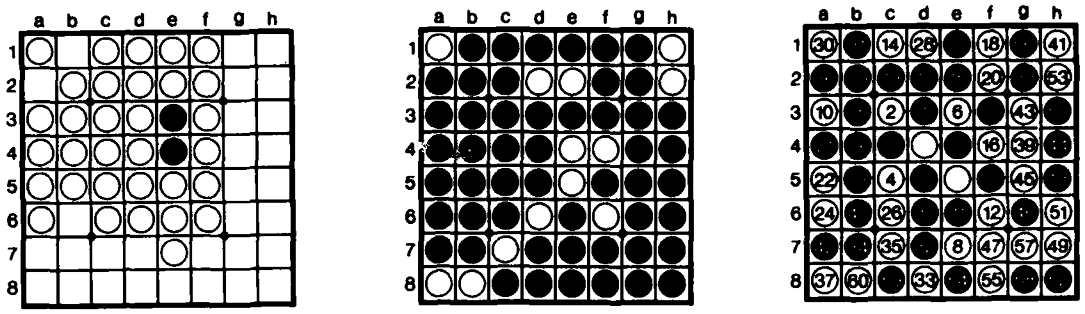
Play the move which captures the most discs

As an evaluation function for a program, this heuristic is generally implemented by maximizing the difference in the number of discs for each side. This will be positive when the player has more discs, negative if the opponent has the majority, and zero for a tie. One of the intriguing aspects of Othello is that this obvious strategy is a miserable failure. Just how wrong this strategy can be is well documented by a game played at the Santa Cruz Open by ***Iago*** against ***The Moor*** (written by David Levy and Kevin O'Connell). Figure 3-1a shows the state of the game after move 30 (Black to play). ***Iago*** was playing black and ***The Moor*** was playing white. At that point ***The Moor*** had a 30 disc advantage (32 to 2). Its problems are two-fold: (1) white has the most discs, but nothing prevents black from recapturing them; and (2) white has completely lost its *mobility*. In its most general sense, mobility refers to the amount of freedom a player has in the selection of moves. In this case, black should be able to force white to move wherever he pleases. That is precisely what happened; leading to a final score of 51-13 in favor of ***Iago*** (Figure 3-1b).

Figure 3-1c shows the record of the game in Othello notation. The numbered discs show when, and by whom, a disc was played in the associated square. For example, the first move was made by black at *d3*, so

[5]The author is not a serious Othello player and except briefly at tournaments expert players have not been available for consultation

there is a black disc, with the digit 1 superimposed on it at *d3*. Each square in the record has a colored disc and a move number (between 1 and 60). Figure 3-2 tracks the disc differential over the whole game. Though ***The Moor*** was probably not playing a pure maximum disc strategy, its strategy was greedy enough to show that the maximum disc strategy is poor.



(a) After the 30th move
Black to play

(b) The final position

(c) The game record

Figure 3-1: ***Iago*** (B-51) vs. ***The Moor*** (W-13)
Santa Cruz Open Machine Othello Tournament

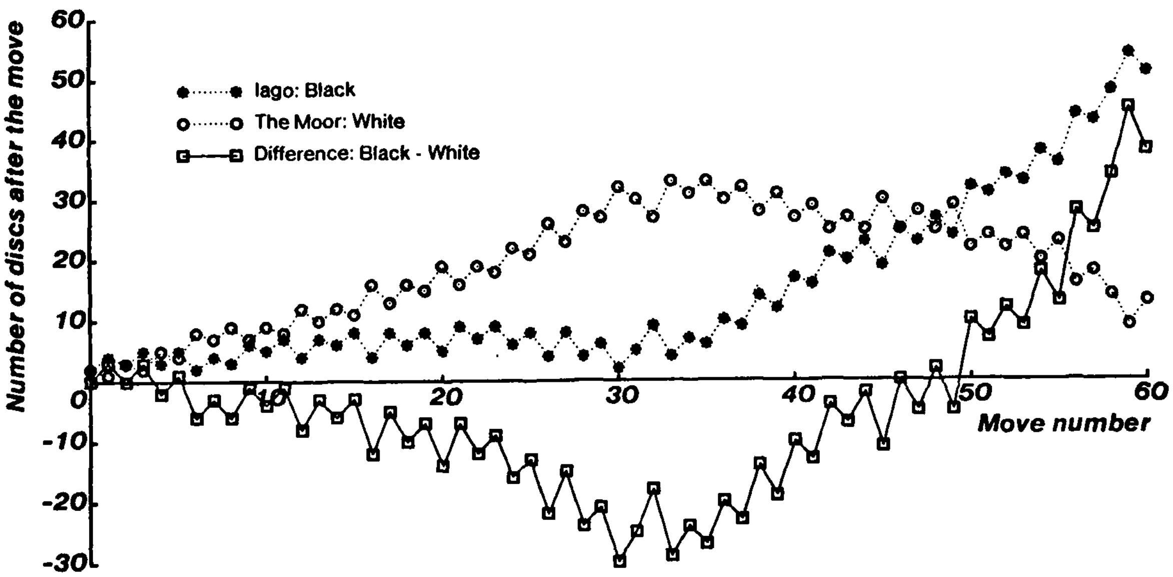


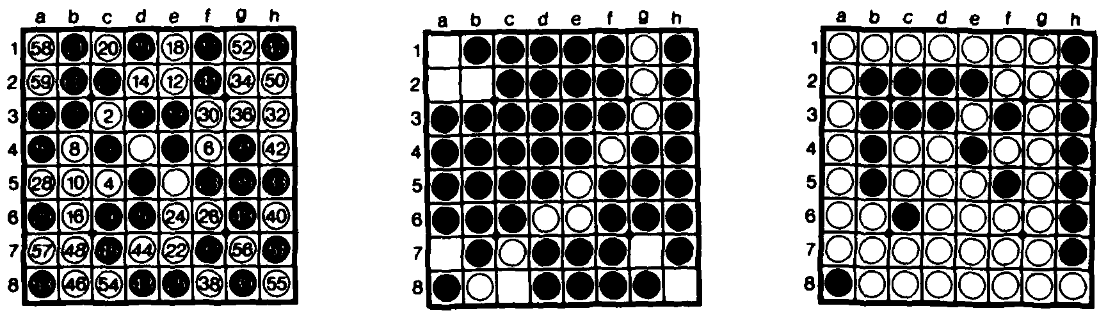
Figure 3-2: The disc counts and differential
Iago (B-51) vs. ***The Moor*** (W-13)
Santa Cruz Open Machine Othello Tournament

3.1.2. The weighted square strategy

The *weighted square* strategy stems from the observation that not all of the squares on the Othello board are of equal value. For example, compare the corner squares (*a1, h1, a8, h8*) with the x-squares (recall Figure 2-1b). The corner squares, once occupied, cannot be recaptured, while ownership of an x-square almost always allows the opponent to capture the adjacent corner. The corner square thus seems a more valuable square.

The maximum disc strategy ignores these distinctions; it just computes the numerical difference in disc ownership. With the weighted square strategy, a sum is computed in which each disc is weighted according to the value of the square that it occupies. The difference between the two weighted sums (one for each player) is the important statistic. The square values can be constants, or can change dynamically with the game situation, depending on the sophistication of the implementation.

The version of ***Iago*** that competed in the Northwestern Man-Machine Othello Tournament played a weighted square strategy (see Maggs (1979) for a description of a typical weighted-square program). Its performance in that tournament revealed the inadequacy of the strategy. ***Iago*** was soundly beaten by both human entrants. ***Iago***'s game against Hiroshi Inoue (won by Inoue 43-21) can be seen in Figure 3-3a. The weighted square strategy fails because there are more important reasons for taking (or avoiding) squares than just their location on the board. For instance, mobility is crucial. Starting from the position in Figure 3-3b, Inoue was able to make six consecutive moves: *c8, h8, g7, a7, a1,* and *a2*. ***Iago*** had no legal moves during that entire span. The final board position can be seen in Figure 3-3c.



(a) The game record

(b) White gets next six moves *c8, h8, g7, a7, a1*, and *a2*

(c) The final position

Figure 3-3: ***Iago*** (B-21) vs. Inoue (W-43)
Northwestern Man-Machine Othello Tournament

The weighted square strategy is not all bad. Against the other five programs in the Northwestern tournament, *Iago* ended up a respectable 3-1-1. The strategy is not, however, sufficient to play Othello at a high level of skill.

3.1.3. The minimum disc strategy

The remaining simple strategy that is worth examining is the minimum disc strategy. Both the maximum disc and weighted square strategies ran into mobility problems because of their greediness. The *minimum disc* strategy attempts to alleviate this by taking the opposite tack. A player using the minimum disc strategy minimizes the number of his own discs, while maximizing the number of his opponent's discs.

I have no direct experience with this strategy. It is an approximation to the mobility measures that will be discussed shortly, but its emphasis is wrong. Where the discs are placed is as crucial as the simple count of how many there are. Having only a few poorly placed discs can be disastrous, as the position in Figure 3-4 shows. Black has fewer discs (7 to 39), but only two moves -- both of which allow white to take corner squares. This example is from Stringham (1980), which contains a discussion of the disc counting strategies.

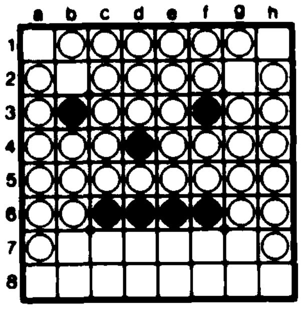


Figure 3-4: A board position that is misjudged by the minimum disc strategy -- Black to play. This position is taken from Stringham (1980)

3.2. General analysis

In a game of Othello, the players must maintain two top-level goals for their play. One top-level goal is obvious: to win the game. The other top-level goal, maximizing the winning margin, is important during tournaments in which total disc differential is used as a tie breaking mechanism. Disc differential is also the primary factor in the official ratings of the United States Othello Association (USOA). A player's rating will be lowered, even if he wins, if he does not beat his opponent by an amount determined by their relative ratings[6].

As with many other games, Othello games can be temporally subdivided into three phases: (1) the opening game; (2) the middle game; and (3) the end game. There are no firm boundaries between the phases. Rather, they are characterized by their different strategic concerns. In the remainder of this section we examine the three phases of an Othello game in reverse chronological order. The analyses of the early phases of the game are motivated by what comes later in the game.

3.2.1. The end game

As the end of the game nears, the primary concern is with maximizing the disc differential. However, as was shown by the failure of the maximum disc strategy, flipping a disc does little good if the opponent can immediately flip it back. The concern must be with maximizing the *final* disc differential.

3.2.1.1. Solving end-game positions

One way to maximize the final disc differential is by solving the position; that is, to completely search the game tree to the end of the game. In Othello, the branching factor of the game tree is always bounded above by the number of empty squares on the board, which diminishes as the end approaches. Figure 3-5 shows the average number of moves available at each point in the game. The data is averaged over the ten games played by ***Iago*** at the Santa Cruz Open (includes two unofficial games also won by ***Iago***), using the values for both players. Pairs of adjacent moves (the first move for each player (moves 1 and 2), the second move for each player (moves 3 and 4), etc.) were then averaged to yield the points in Figure 3-5. Because of the small branching factor near the end, complete analysis of final move sequences plays an important part in successful end game play.

[6]The ratings are defined so that every ten points difference between two players means one disc difference in score. For example, a game between two players whose rankings differ by 200 points (the interval between classes of players) should end with a disc differential of 20 (a final score of 42-22). Ratings below 800 points are in the novice class; over 1999 points is a senior master. The highest ranking (1796 -- high expert) currently belongs to Jonathan Cerf. For more on the ratings system see Richards (1981).

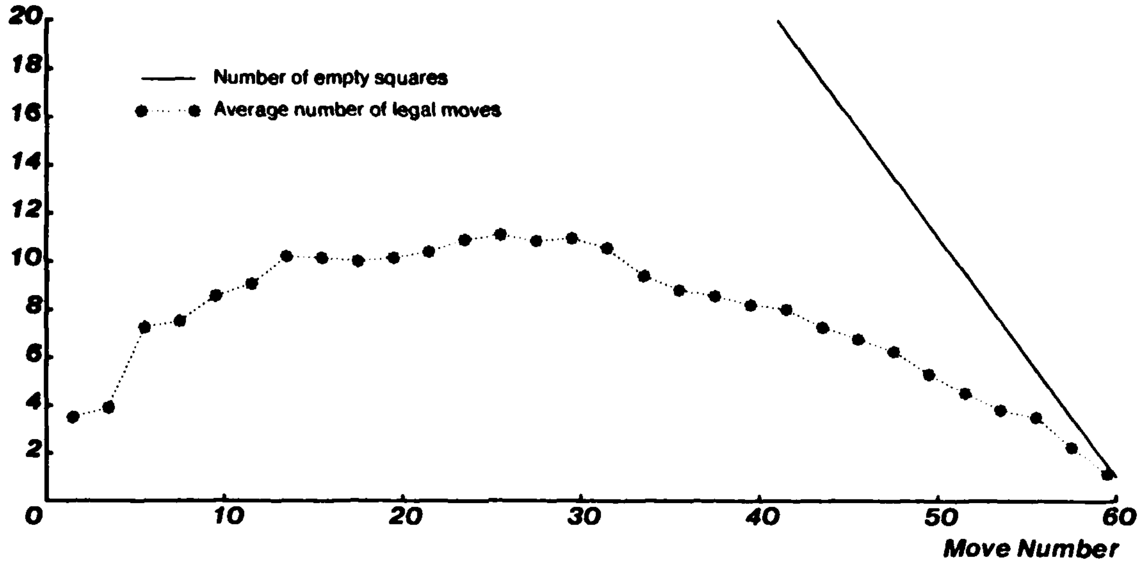


Figure 3-5: Mean number of legal moves at each point of an Othello game. Computed from the ten games played by ***Iago*** at the Santa Cruz Open

3.2.1.2. Stability

Prior to when complete analysis is possible, possession of final discs can be guaranteed through the acquisition of *stable* discs. A completely stable disc can never be recaptured (flipped) by the opponent. Therefore, the number of stable discs for each player is a monotonically increasing function of move number. Figure 3-6 shows the number of stable discs for the game at the Santa Cruz Open between ***Iago*** and ***Reversi Challenger*** (written by Dan and Kathe Spracklen). The game (won by ***Iago*** by a score of 60-3) can be seen in Figure 3-7.

Four lines run through each square on the Othello board (one horizontal, one vertical, and two diagonals). Two necessary (but not sufficient) conditions for flipping a disc are that there be one of the player's discs on one side of the desired disc (on one of the four lines through the square), and a blank square on the other side (along the same line). Neither condition requires immediate adjacency (there may be intervening discs of the same color as the one being flipped), but this bracketing along one of the lines must be possible. Therefore, a disc is provably stable if there are no lines through the square along which both of these conditions can ever hold.

The Othello board can be divided into three classes of squares: (1) *corner* squares (*a1*, *h1*, *a8*, *h8*); (2) *edge* squares (the a-squares, b-squares, and c-squares in Figure 2-1b) ; and (3) *internal* squares (the remaining

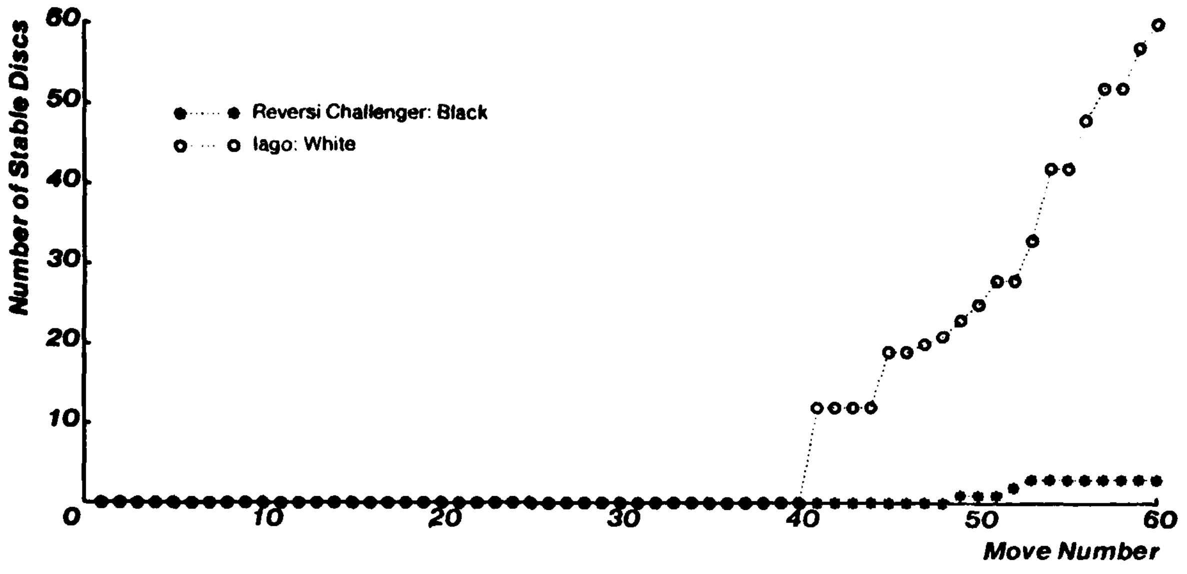
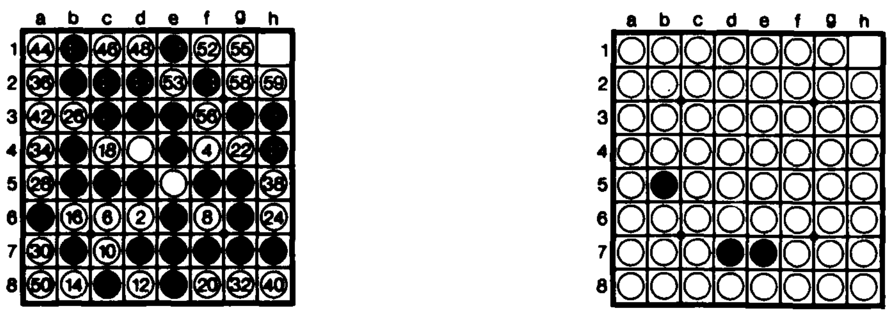


Figure 3-6: The number of stable discs
Reversi Challenger (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Computer Othello Tournament



(a) The game record

(b) The final position

Figure 3-7: ***Reversi Challenger*** (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Machine Othello Tournament

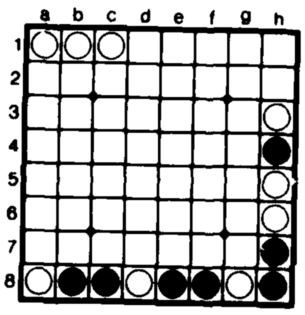
squares). Discs in corner squares are always completely stable. At least one side of each line through a corner square lies off the playing surface, inaccessible for disc placement. Corner discs are crucial, not only because of their own stability, but because they commonly form an anchor by which other discs can be stabilized. Edge discs are the first to show this effect. Three of the four lines through each edge square lie off the board on one side, leaving only one line of possible instability -- the edge itself. One way of removing this instability is to have the edge disc be immediately adjacent to a stable disc of the same color, such as a corner disc. If there is no way that the stable disc can be flipped along the edge, the same must be true of an adjacent edge disc of the same color. In general, there are three types of formations that stabilize edge discs.

- If the edge disc is adjacent to a stable edge (or corner) disc of the same color then it is stable. For example, white's discs at *b1* and *c1* in Figure 3-8a.

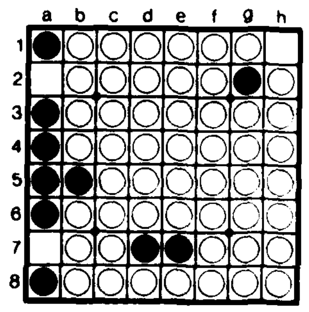
- If the edge disc is part of a string of the player's discs embedded between two of the opponent's stable discs, then it is stable. The embedded disc can also be stable even when the embedding discs are not; though black's disc at *h4* is unstable (Figure 3-8a), white's discs at *h5* and *h6* are stable. All of the empty squares in the top portion of that edge are needed just to flip black's disc, leaving no possibility of flipping white's discs.

- If there are no empty squares on that edge (including corners) then all of the discs on that edge are stable. For example, all of the discs on the bottom edge of the board in Figure 3-8a are stable.

In addition to these stable formations, it is possible (though rare) for an edge disc to be stable simply because it is impossible for the opponent to make the edge move that would result in the disc being flipped. In Figure 3-8b, the black discs at *a3*-*a6* are stable because it is impossible for white discs to be played at *a2* and *a7*.



(a) *b1*-*c1*, *h5*-*h7*, and *b8*-*g8* are stable.

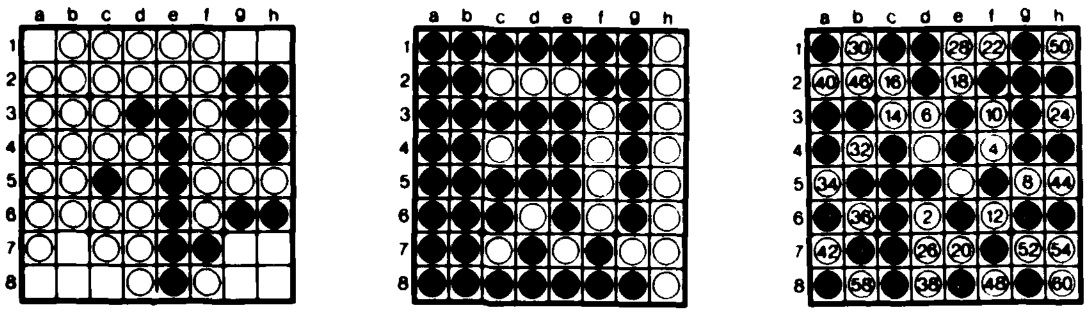


(b) *a3*-*a6* are stable (among others)

Figure 3-8: Contrived examples of edge stability.

Internal discs require stability along all four lines through them, and are thus difficult to stabilize until very late in the game. The maximum disc strategy ignores this fact (among others), and plays poorly. On the other hand, the weighted square strategy is a first-order attempt at handling the stability distinctions between the regions of the board, with weights diminishing from corners to edges to internal squares. Of course, this approach is too simplistic; the impact of owning a disc can extend far beyond the square occupied by the disc.

For example, placing a disc in a corner square may allow a string of discs along one side to be stabilized (along with stabilizing the corner square itself), while wreaking havoc along other sides of the board. Figure 3-9a shows a typical such situation, from the first game of the 1980 world-championship match between Jonathan Cerf (U.S.A) and Takuya Mimura (Japan). White (Mimura) has an *unbalanced* formation [Jacobs & Jacobs 79] along the top edge of the board -- a string of five discs on the side, with the corners and one of the c-squares empty. A move to *h1* by white stabilizes white's discs along the right side of the board, but allows black to stabilize the top and left sides. In addition, black gets the corner at *a8*, giving him a leg up on the bottom edge of the board. The results can be seen in the final board position (Figure 3-9b). The game record is in Figure 3-9c.



(a) White to play (Before Move 50)

(b) The final position

(c) The game record

Figure 3-9: A case where the corner move (50-h1) leads to bad consequences
First game of the 1980 World Championship match
Cerf (B-44) vs. Mimura (W-20)

Strategies for occupying and avoiding specific squares become a major component of play when the effect of a single disc is critical. A player can have one (or more) of the following goals for a square: (1) occupy the square himself; (2) avoid the square; (3) have the opponent occupy the square; and (4) keep the opponent out of the square. Each type of goal requires a different strategy, but they are all based on a common set of four components:

- ***Work on the complementary goal.*** For example, if a specific square is desired, the complementary goal of keeping the opponent out of the square should also be pursued.

- ***Work on ownership of the adjacent squares.*** If the player has discs in the squares adjacent to the contested square, the opponent is likely to get the square (and vice-versa).

- ***Look for sequences of moves with the desired result.***

- ***Maximize the player's mobility, and minimize the opponent's mobility.*** A player with low mobility can be forced into taking undesirable squares. A player with high mobility has a better chance of finding sequences of moves with the desired result.

3.2.2. The middle game

During the middle game, *mobility* -- flexibility in the choice of moves -- is the critical strategic concern. Lack of mobility can lead a player into two distinct difficulties. The game at the Santa Cruz Open between ***Iago*** and ***The Moor*** (recall Figure 3-1) illustrates one type of difficulty. ***Iago*** took advantage of the early loss of mobility by ***The Moor***, by consistently leaving ***The Moor*** with a choice among a small set of moves. When possible, the set included only bad moves, resulting in a gradually deteriorating position for ***The Moor***. Figure 3-10 shows the number of legal moves for each side in the game.

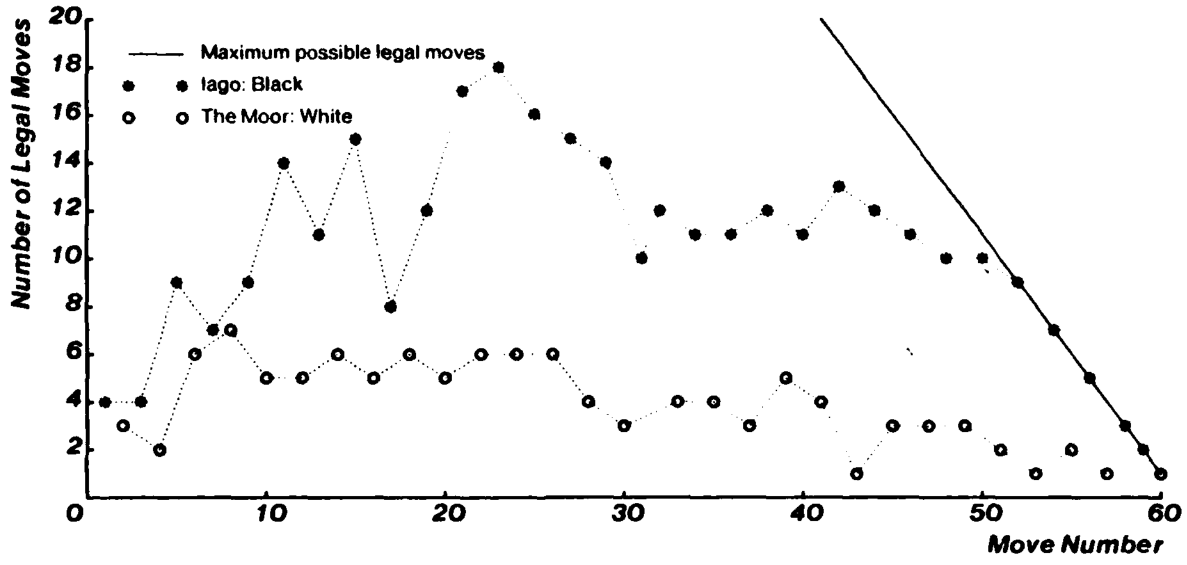


Figure 3-10: The number of legal moves
Iago (B-51) vs. ***The Moor*** (W-13)
Santa Cruz Open Machine Othello Tournament

As an alternative to using a mobility deficiency to force bad moves, the opponent's moves can be eliminated entirely. The game at the Northwestern tournament between ***Iago*** and Hiroshi Inoue (recall Figure 3-3) illustrates how this can lead to the second difficulty. ***Iago***'s mobility was restricted for most of the latter part of the game. Inoue took advantage of this by making six of the last seven moves in the game, generating a large number of stable discs at the last minute.

Whichever tack is taken, the development of a mobility advantage becomes a crucial strategic concern in the middle game. However, it is clear that this involves more than just maximizing the difference in the current number of moves available to the players. First, the desirability of the moves matters. A player who has a large number of moves, all of which are undesirable, suffers from the first difficulty described above -- he must make bad moves. The other problem with the simple notion of mobility is that, in addition to immediate mobility, the players must consider what other features of the board will lead to a long-term mobility advantage. In all, there are three aspects of the board that impact mobility.

- *The current number of acceptable*[7] *moves.* The difference in this value for the two players determines who has the immediate mobility advantage. In addition, all else being equal, the advantage will propagate some distance into the future, impacting future mobility.
- *The current number of unacceptable moves.* While of little use in the immediate situation, given time, unacceptable moves can become acceptable. For example, an x-square move becomes acceptable once either the adjacent corner is occupied, or it no longer matters if the opponent takes the corner, or the opponent has no way of flipping the x-square disc along the diagonal (if he has no discs on the diagonal, and no way of getting any). This proves to be a common way of squeezing out a last one or two reasonable moves.
- *The mobility potential of other board configurations.* According to the rules of Othello, a formation consisting of (1) an empty square, followed by (2) a solid string of the opponent's discs, followed by (3) one of the player's discs, is required for a legal move in the empty square. By working on each aspect independently, at some later point there should be a payoff in terms of an additional advantage in the number of moves. For example, one aspect involves discs that are adjacent to empty squares (*frontier* discs). Acquisition of frontier discs limits the long-term mobility of the player while allowing the opponent to increase his mobility. Figure 3-11 shows an example from the game at the Santa Cruz Open between ***Iago*** and ***Reversi Challenger***. ***Reversi Challenger*** has built a *wall* of frontier disks along the top, disallowing any moves by it in the top half of the board until after ***Iago*** breaches the wall. ***Iago***, on the other hand, has its choice of moves across the wall.

3.2.3. The opening game

The opening game should be based upon a *book* containing successful opening sequences of moves, but little has been published (at least in English) on the subject[8]. However, there is agreement on the first two moves in the game. The first move (by Black) is a choice between four symmetric moves (*d3*, *f5*, *e6*, and *c4*); the choice is arbitrary. White selects the initial line of play with his first move (the second move in the game). He has three alternatives: (1) the *perpendicular* opening (Figure 3-12a); (2) the *diagonal* opening (Figure 3-12b); and (3) the *parallel* opening (Figure 3-12c). The perpendicular and diagonal openings show up in expert Japanese play [Albion 80]; the parallel opening does not.

[7]Rigorously defining "acceptable" is problematic; it is used loosely here to refer to moves that do not have disastrous consequences.

[8]Albion (1980) contains a description of one opening (the Maruoka opening), and some variations.

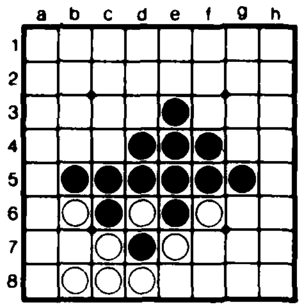
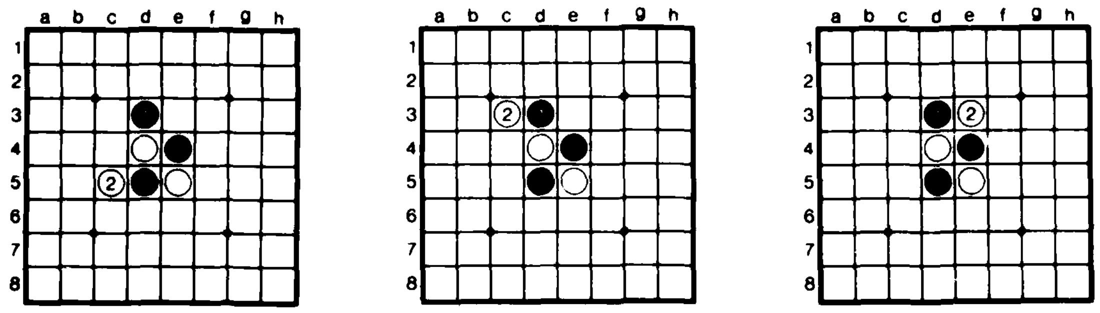


Figure 3-11: A wall trapping black in the lower half of the board
Reversi Challenger **(B-3)** vs. ***Iago*** **(W-60)**
Santa Cruz Open Machine Othello Tournament



(a) The perpendicular opening **(b)** The diagonal opening **(c)** The parallel opening

Figure 3-12: Initial game records for white's three opening moves

4. Iago

The primary task faced by ***Iago*** is to select the best move, whenever it is its turn to play. To accomplish this, ***Iago*** is built around α-β search (see Nilsson (1971)), and an evaluation based on the Othello knowledge described in Section 3. These two components form the heart of ***Iago*** (and this section), but they are not sufficient alone because of the time constraints under which Othello games (at least during tournaments) are played. Care must be taken so that ***Iago*** can complete all of its moves within its allocated time (30 minutes in tournament games).

4.1. Time allocation in Iago

During tournament matches, ***Iago*** must allocate its 30 minutes effectively. This would yield one minute per move if it were spread equally over all of the moves (assuming that ***Iago*** makes half of the moves). Five factors make this simple approach inadequate:

1. Time that is left unused by one move selection can be fruitfully added to later moves.
2. There is overhead time for disc flipping (on the physical board), typing in moves, and hitting the button on the chess clock. This time is charged to ***Iago***, in addition to the time taken to select a move.
3. Some time must be left in case hardware problems arise during the match. Tournament rules vary, but computer downtime is frequently charged to the program's clock.
4. ***Iago*** may have to make more than the standard 30 moves (in case its opponent can't move at some point in the game).
5. The value of additional clock time varies across moves. For example, it can pay to allocate more time to end-game moves if that allows the positions to be completely solved.

Before beginning each move selection, ***Iago*** generates a time allocation for the selection. This allocation is computed by multiplying the amount of time ***Iago*** has remaining on its clock by a pre-computed allocation factor for the current point in the game (move number). These factors were computed by the following procedure:

1. A marginal value was assigned to each move number, reflecting the importance of that point in the game.
2. The values were then normalized so that the odd and even fractions would separately sum to 1 (corresponding to plays by black and white respectively).
3. An allocation fraction for each move number was then computed as the ratio of its marginal value to the sum of the marginal values from there to the end of the game.

4. Some fractions were then reduced by hand to leave room for more than 30 moves[9].

The resulting normalized marginal allocations are shown in Figure 4-1. The raggedness near the end of the curve is a result of the hand modifications. The first move has an abnormally small allocation because it is a choice between four symmetric moves. Any time spent on this selection is a waste, and there is no opening book in which to prestore a selection.

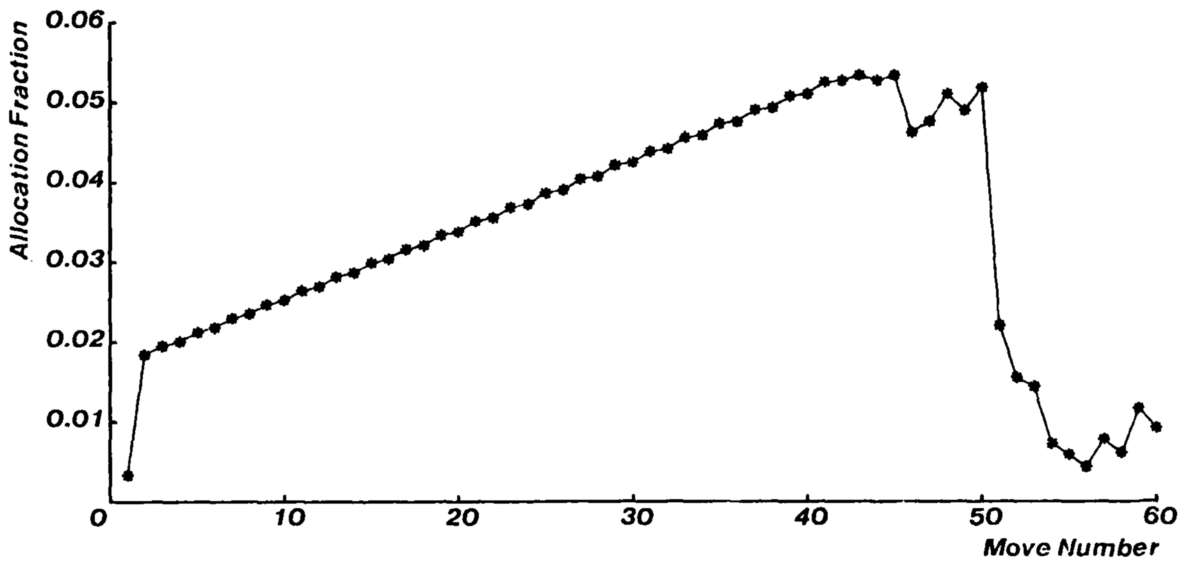


Figure 4-1: Marginal time allocation fractions (normalized) by move number

The time allocation is generated by multiplying the amount of time left on the clock by the allocation fraction. This procedure takes care of points 1, 4, and 5. Potential hardware problems (point 3) are alleviated by removing 2.5 minutes from ***Iago***'s internal clock at the beginning of the match, leaving it with only 27.5 minutes available for allocation. Overhead time (point 2) is handled by subtracting 10 seconds from the time allocated for the move selection. These 10 seconds are then used by the overhead activities instead of by the move selection procedure.

[9]This was a post-hoc solution that became necessary during the Santa Cruz Open. There were several games in which ***Iago*** nearly ran out of time (using 28 of its 30 minutes).

4.2. Search in Iago

The basic search procedure in ***Iago*** consists of an implementation of a full-width recursive α-β algorithm. The root node of the search tree returns the move that must be made in order to reach its best (determined by the backed-up evaluation) child node (the *first-ply* nodes). The remaining nodes back up the evaluation from the terminal nodes of the tree.

Iterative deepening [Slate & Atkin 77] is used to determine the depth to which ***Iago*** can search during the time allocated. ***Iago*** performs a complete 1-ply search, followed by a complete 2-ply search, and so on. ***Iago*** tries another ply when it estimates that it can completely search at least Min{3, number of first-ply nodes} of the first-ply nodes at the new depth. In order to make this estimation, ***Iago*** keeps track of the amount of time spent for each iteration, and the average branching factor for each player. Using this information, it extrapolates the expected time to complete another iteration.

Even when ***Iago*** estimates that it can finish another iteration, the amount of time required to finish might far exceed the time allocated for that search. To handle this, ***Iago*** checks its progress between each first-ply node, and aborts the search if it estimates that the time to complete one more first-ply node will exceed 1.3 times the allocated time. During each search, the first-ply nodes are ordered according to their values. Because this ordering is used to establish the search order for the next iteration, even if the search is aborted after only one first-ply node, it is safe to pick the best move found during the current iteration.

Iago currently searches to an average depth of 6.3 (i.e. searches are completed for .3 of the first-ply nodes at a depth of 7) . Deeper searches do result in better performance, but increasing the depth of search is not a panacea. In the Northwestern tournament, the two worst programs searched to an average depth of 7-8 ply, while the better programs searched to only 4 or 5 ply.

4.2.1. Increasing the efficiency of the search

Iago's search depth has been increased by: (1) upgrading the efficiency of the evaluation function; (2) performing all computations as close to the root node as possible (*incremental updating of the evaluation function*); and (3) decreasing the number of nodes that the search must examine. The techniques included in (1) are discussed along with the evaluation function itself in Section 4.4.

Because each additional ply in the search adds a loop nested within the current innermost loop of the computation, technique (2) is equivalent to the optimization technique of moving code out of loops. Many of the values that are needed in the evaluation function can be computed by initializing them at the beginning of the search, and updating them as the search proceeds. This procedure takes computations that would otherwise be done completely at the terminal nodes of the search (inside the innermost loop), and spreads them out over the entire search tree (moving much of the effort out of the inner loops).

The third technique is where the α-β algorithm is so important. With a 30 minute time allocation in which to make its moves (tournament situation, but with no overhead charges), ***Iago*** examines an average of 10000 terminal nodes per move[10] (counting the terminal nodes from all iterations of the search). The average raw branching factor is 9.9, so with a simple minimax search (no iterative deepening), ***Iago*** could only achieve a search depth of 4.0. ***Iago*** achieves its average search depth of 6.3 by lowering the branching factor to 4.0. The minimal attainable branching factor for α-β is 3.7 (see Slagle & Dixon (1969) for the defining formula).

Since the order in which nodes are searched is crucial to the efficiency of the α-β algorithm, ***Iago*** devotes considerable resources to improving on random ordering. ***Iago*** gathers information about node ordering during iterative deepening. The top three ply of the search tree are ordered and saved once they have been searched -- guaranteeing perfect ordering (according to the information obtained so far) for the critical top portion of the tree. Below the top three ply, ***Iago*** maintains a *response killer table* (similar to the *refutation data* used by Cichelli (1973a)). For each possible move (combination of square and player), the table contains a list of all possible responses, and a rating for each response. The ratings are computed incrementally over iterations of the search (and across successive moves in the game, though they are halved between moves to emphasize the effect of killers found on the current move) by boosting the response value by a small amount whenever it is a legal response, and by a large amount whenever it is the best response (or the cause of a cutoff in the search). Therefore, ***Iago*** can generate an ordering at each node by looking at the ratings of the responses to the move made at that node.

A two-factor experiment (depth to which the search tree was saved versus the ordering heuristic used for the remainder of the search tree) was performed to evaluate the effectiveness of these techniques (see Gillogly (1978) for an analysis of the effectiveness of search heuristics in chess). Three different depths were used: 0, 1, and 3. In the remainder of the tree, four ordering heuristics were tried: (1) a fixed ordering of the nodes based on the "natural" order of move generation (left-to-right across the board and top-to-bottom); (2) a fixed ordering according to the "value" of the squares (e.g. corner squares are first); (3) a killer table based on who is to move, and the move number (similar to the *killer heuristic* first suggested by McCarthy in 1957 [McCarthy 81]); and (4) the response killer table.

The twelve conditions (3 × 4) were all evaluated on the same set of six data points selected from five games at the most recent world championship tournament [Prentice 81]. Two points were selected randomly from moves 1-15 of the five games, two from moves 16-30, and two from moves 31-45. For each point, the search depth was set so that the standard condition (tree depth of 3, response killers) would use about 1 minute. This

[10]This value (as well as the value for the branching factor) was computed by averaging the values over the first 46 moves of a game played by ***Iago*** against itself (end-game searches were done from move 47 until the end of the game (see Section 4.3)).

depth was then used for all twelve conditions. The mean (and standard deviation) of the branching factor for each condition (averaged over the 6 points) can be found in Table 4-1. The minimal obtainable branching factor for the six points is 3.9 (standard deviation of 0.8). The best condition (search tree saved to a depth of 3, with the response killer table used in the remainder of the tree), is the one used in ***Iago***. This still leaves room for improvement, as the branching factor for this condition (4.1) differs significantly from the optimal value ($p < 0.05$).

Depth Saved	Ordering Heuristic				Marginal
	Fixed Order		Killer Table		
	Move Generation	Square Value	Move Number	Response	
0	5.6 (1.3)	5.4 (1.8)	4.4 (0.8)	4.2 (0.8)	4.9 (1.3)
1	5.5 (1.2)	5.1 (1.6)	4.5 (0.8)	4.3 (0.9)	4.8 (1.2)
3	4.7 (0.9)	4.4 (1.0)	4.2 (0.7)	4.1 (0.8)	4.3 (0.8)
Marginal	5.2 (1.1)	4.9 (1.5)	4.4 (0.7)	4.2 (0.8)	4.7 (1.1)

Table 4-1: Evaluation of α-β ordering heuristics in ***Iago***.
Three depths to which the search tree is saved × four ordering heuristics
Mean (and standard deviation) of the branching factor for each condition.

Looking at the marginal means, there is a clear trend along both dimensions: the more ordering information available, the lower the branching factor. Along one dimension, as the depth to which the search tree is saved increases, the branching factor decreases. Along the other dimension, both fixed ordering schemes maintain a single ordered list of squares (the square-value list has effectively more information because the ordering is based on knowledge of the game), while the move-number killer table requires up to 15 (the maximum search depth) lists, and the response killer table requires up to 60 (the number of squares in which it is possible to play) lists.

While the trends are clear, not all of the results achieve statistical significance[11]. There is a significant difference between tree depths of 1 and 3 ($p < 0.001$), but the difference between 0 and 1 did not achieve significance (though $p < 0.1$ does hold). Along the other dimension, the two fixed ordering schemes do not significantly differ ($p < 0.15$), and the same holds for the two killer tables ($p < 0.1$). When the fixed ordering data is pooled and compared against the pooled killer data, the difference is significant ($p < 0.005$).

[11]The data were normalized before they were checked for significance, by dividing each value by the branching factor for the control condition (none of the search tree was saved, and the move-generation fixed ordering was used).

4.3. Solving end-game positions

Iago is capable of solving (through a complete search to the end of the game) a significant number of end game positions. In this context, solving a position can have two distinct meanings, corresponding to the two top-level goals in Othello (Section 3.2): (1) to find *the* move which maximizes the final disc differential; or (2) to find *any* move which yields the best possible value from the set {*win, tie, loss*}. ***Iago*** has a distinct (from each other, and from the evaluation function used in the earlier part of the game) evaluation function for each of these goals, consisting of the obvious measure for each interpretation: (1) the final disc differential; and (2) a three valued measure (from the set {*win, tie, loss*}).

Evaluation (1) is logically sufficient for both top-level goals, but evaluation (2) can be employed earlier in the game. ***Iago*** is capable of performing a complete end-game search with evaluation (2) at around move 46 (15 moves from the end of the game). Employing evaluation (1) is usually feasible at move 48 (13 Moves from the end). Two aspects of Evaluation (2) are responsible for its ability to be used prior to Evaluation (1). Because evaluation (2) can return only one possible value for a winning position, cutoffs abound, once the first winning path has been found in the search tree. In addition, the search order when evaluation (2) is employed should be nearer to optimal, as it is only necessary to assure that the winning moves precede the ties and losses. Extra search is not required just because the best winning move was not searched first.

Iago decides dynamically which evaluation to use. At every iteration in the search, a decision is made whether to go one deeper with the normal evaluation, use evaluation (1), use evaluation (2), or terminate the search. The decision as to which search to perform, is based on empirically determined equivalences in search time. Using either evaluation (1) or (2) the search can get deeper than is possible with the normal evaluation because of the smaller time costs of the evaluation functions involved. If the search depth for the current iteration is n, the next iteration will be either a search with the normal evaluation to a depth of $n+1$; a search with evaluation (1) to a depth of $n+5$; or one with evaluation (2) to a depth of $n+7$. The search times for these alternatives are approximately equal. The decision algorithm is:

```
If there is not enough time for another iteration
        Then terminate the search
Else If the current position is within $n+5$ moves of the end of the game
        Then use evaluation (1)
Else If the current position is within $n+7$ moves of the end of the game
    And there is not enough time to do two more iterations and evaluation (1)
        Then use evaluation (2)
Else complete another iteration with the normal evaluation.
```

4.4. Iago's evaluation function

When ***Iago*** is unable to search to the end of the game, it makes use of a single evaluation function consisting of four components based on the analysis of Othello strategy in Section 3. The components are weighted by *application coefficients* [Berliner 80], and then summed to yield a single value for the evaluation. The coefficients used in ***Iago*** were based initially on opinions about the relative importance of the individual components. A limited set of variations on these values was then evaluated by playing these differing versions of ***Iago*** against each other. The best variation became ***Iago***'s evaluation function[12]:

$$\begin{aligned}\text{Eval(pos)} = {} & \text{ESAC}(\mathit{MoveNumber}) \times \mathit{EdgeStability} + 36 \times \mathit{InternalStability} + \\ & \text{CMAC}(\mathit{MoveNumber}) \times \mathit{CurrentMobility} + 99 \times \mathit{PotentialMobility}\end{aligned}$$

Two of the application coefficients vary with move number to reflect the relative importance of those components during different stages of the game.

$$\text{ESAC}(\mathit{MoveNumber}) = 312 + 6.24 \times \mathit{MoveNumber} \qquad 1 \leq \mathit{MoveNumber} \leq 60$$

$$\begin{aligned}\text{CMAC}(\mathit{MoveNumber}) &= 50 + 2 \times \mathit{MoveNumber} && 1 \leq \mathit{MoveNumber} \leq 25 \\ &= 75 + \mathit{MoveNumber} && 25 \leq \mathit{MoveNumber} \leq 60\end{aligned}$$

Notice that the edge-stability application coefficient is almost an order of magnitude greater than any of the others. This insures that nontrivial edge-stability values will dominate the values from the other components, moving ***Iago*** automatically into the end-game.

The value for ***Iago***'s evaluation function (on its moves) in the game at the Santa Cruz Open vs. ***Reversi Challenger*** is shown in Figure 4-2. These values were determined by a post-hoc analysis of the game record[13]. The graph terminates (after move 46) when ***Iago*** is able to exactly solve an end-game position (Section 4.3).

The remainder of this section is devoted to descriptions of the four components of the evaluation and a brief discussion of the opening game, a weakness in ***Iago***.

4.4.1. Stability

Iago's evaluation function has two stability components: (1) *EdgeStability* -- the corner and edge squares, and their interactions with x-squares; and (2) *InternalStability* -- the stability of non-edge squares.

[12]To facilitate comparisons in this presentation, the component ranges have all been normalized to [-1000, 1000], with the differences moved into the application coefficients.

[13]***Iago*** is set up to compute statistics for each position that occurred in a game. Tournament time allocations are set (except that the 10 seconds per move for overhead is not charged), and a search is performed at each position. During the search, ***Iago*** saves information such as the value returned by the search, the move that ***Iago*** would have made, and the values of the components of the evaluation.

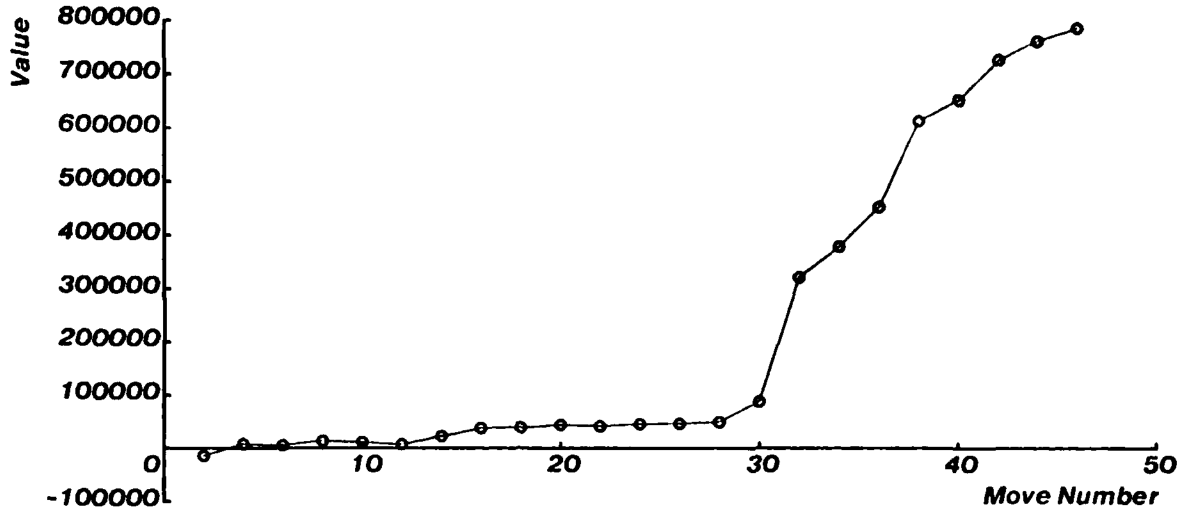


Figure 4-2: *Iago*'s evaluation function on its moves
Reversi Challenger (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Machine Othello Tournament

4.4.1.1. Edge stability

For computations of edge stability, *Iago* assumes that each side of the board can be treated independently. The 6561 (3^8) possible configurations of eight edge squares (including the two adjoining corners), are handled by a precomputed table. Each element in the table reflects the value of one edge configuration for black, assuming that he has the next move[14]. Values for the three other possibilities (the value for black with white to move, the value for white with white to move, and the value for white with black to move) are all retrieved from this table. The value for white is the negative of the value for black, and the values for when white has the next move are determined from the inverse configuration (all white discs changed to black, and all black to white) with black to move.

The table is precomputed by an iterative algorithm. The first step is to initialize the table with a set of *static* values. Each disc in a configuration has a weight that depends on the square it is in (such as corner or c-square), and the stability of the disc -- either stable, semi-stable (cannot be flipped on the next move), or

[14]This assumption is important because of the asymmetry inherent in many side configurations. Recall Figure 3-3b; because white has the next move, he is able to move to *c8* followed by *h8*, stabilizing seven of the squares along the bottom edge. If black had had the next move, he could have turned it around by moving to *c8*, stabilizing seven of the bottom edge squares for himself.

unstable (Table 4-2). The static value of the configuration is the sum of these weights (positive for black discs and negative for white discs).

	Stable	Semi-stable	Unstable
Corner	700	*	*
C-Square	1200	200	-25
A-Square	1000	200	75
B-Square	1000	200	50

* Impossible case since corners must be stable

Table 4-2: The weights for edge discs.

Each iteration of the algorithm starts with the completely filled configurations (all discs stable) and works backwards to intermediate positions by removing discs from these positions. These intermediate positions are evaluated through the computation of the expected value of the position. For each empty square, the value for when black plays there (computed by playing a black disc and retrieving the value for that position from the table) is multiplied by the probability of being able to make that move. The probability is 1 if black can move there by flipping an edge disc, otherwise it is 0 for corner squares and a function of the neighboring discs for the other six squares. Black is also allowed to avoid making a move on the edge (leaving the edge configuration as it is) with a probability of 1. The best alternative is chosen as the value of the configuration. As values are assigned to intermediate positions, new positions are generated by removing discs from the ones already evaluated. The iteration is complete when a value has been assigned to the empty configuration. The table used in ***Iago*** was the result of five iterations.

The current table is adequate, but overly greedy. An attempt to rectify this problem by using a more sophisticated generation algorithm, led to a table of values that was less greedy, and played a better edge game. Unfortunately, because of a problem to be discussed in Section 4.4.2.2, the non-greediness of this new table caused more problems than it solved.

A more serious problem occurs when the assumption of independence breaks down, primarily at corners. One such problem stems from the table's ignorance of the effect of x-square discs on corner squares (and by extension, to the adjacent edges). Through most of the game, playing a disc in an x-square practically assures the opponent access to the adjacent corner. Though this can greatly affect the edge values, this effect is not handled by the edge table. If corners were always good and x-squares always bad, x-squares could simply be avoided. There are, however, major exceptions to this rule. As the situation in Figure 3-9 shows, black's move to *g2* (an x-square), at move 45, is an excellent move. Following white's corner move at *h1*, black is able to stabilize three sides of the board. Under such situations it is often advisable to make the x-square move. If the opponent is low in mobility, he will be forced to take the corner. At worst, it gives the player one more safe move, which can be critical late in the game.

Iago overcomes this weakness in the edge table by explicitly evaluating the interactions between x-squares and corners (and edges). It compares the expected cost of making an x-square move with the value of not making it. The expected cost is computed by estimating the probability of the x-square being used as an access route to the corner square, and the cost of giving up the corner square. The estimation of the probability (p) breaks down into three cases.

- If the adjacent corner is occupied, the probability is 0.
- If the opponent can immediately flip into the corner by way of the x-square, the probability is assumed to be 1[15].
- Otherwise, ***Iago*** assumes the probability to be a decreasing function of move number ($1 - \mathit{MoveNumber}/120$). As the move number increases, there are fewer squares in which it is possible to play, increasing the difficulty of utilizing the x-square.

The cost of giving up the corner square (assuming it is empty) is computed by performing a small search at those terminal nodes of the regular search in which an x-square is occupied, while the adjacent corner is empty. A disc of opposite color from the x-square disc, is played in the corner square (even if it is not a legal move), and the appropriate edge discs are flipped[16] along the two adjacent edges. The values for these two edges are combined into a single value for the corner. It is assumed that the player with the next move will be able to take his choice of which side to play into; so the value for that side (with that player to move next) is added to the value of the adjacent side (with the other player able to move first). This composite value is compared to the composite value for the existing edge configurations; if the new value (for the player who made the x-square move) is less than the current value, the *expected* value ($p \times \mathit{NewValue} + (1-p) \times \mathit{ExistingValue}$) is used instead of the existing one[17].

[15] Due to an oversight, this condition was not actually tested.

[16] The discs are not actually flipped. Instead, an index (based on the color of the disc to be played in the corner and the current edge configuration) is kept to the new value in the edge table.

[17] There is an additional multiplicative factor in the expected value that corrects for the fact that the effect of a corner disc on an edge value is counted twice when the disc is actually on the board (each edge affects two of the four corner values), but only once when it is placed there during the x-square evaluation.

The four corner values are summed to yield a single value for the edge of the board. Figure 4-3 depicts the trajectory of the edge value for ***Iago***'s game in the Santa Cruz Open against ***Reversi Challenger***. (The value takes off after move 30, because ***Iago*** perceives that it will gain a corner (Figure 4-4)). These are the weighted values determined from ***Iago***'s search.

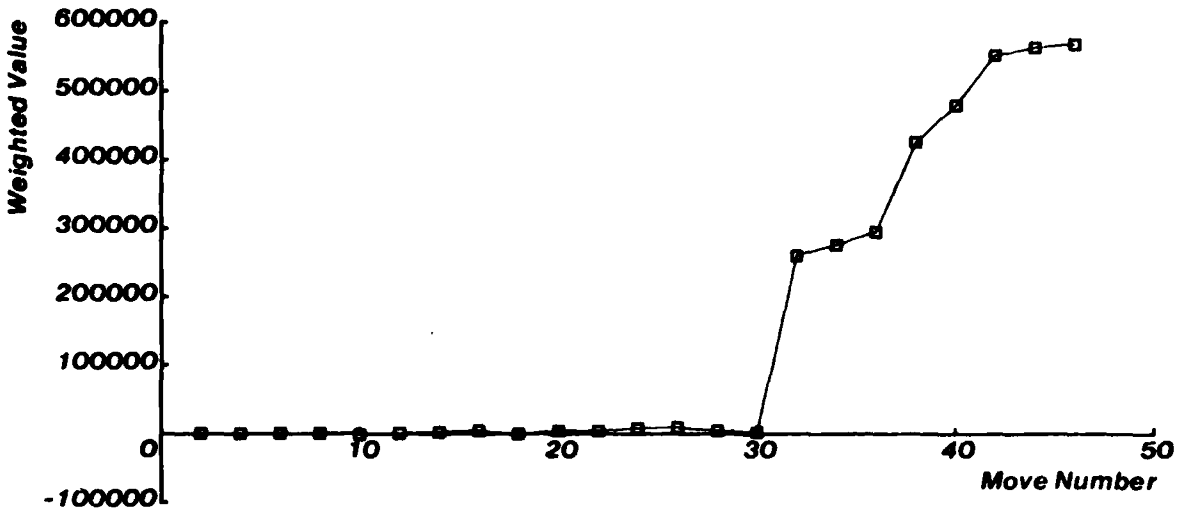


Figure 4-3: ***Iago***'s weighted edge values
Reversi Challenger (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Machine Othello Tournament

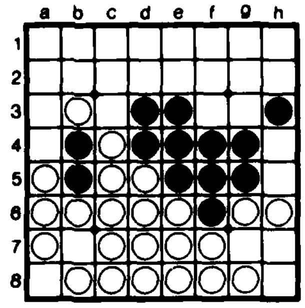


Figure 4-4: After move 32 -- All of black's moves give up a corner
Reversi Challenger (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Machine Othello Tournament

This procedure does a more than adequate job of evaluating interactions between x-squares and edges. For example, in the game at the Santa Cruz Open between ***Iago*** and ***The Moor***, ***Iago*** made three x-square moves -- at moves 29, 34, and 38. Each x-square move sacrificed a corner square, but left ***Iago*** in a position in which it could win handily 51 to 13. However, there are other interactions between adjacent edges that are not handled by either the edge table, or the x-square algorithm. Figure 4-5 shows a contrived situation in which independence fails catastrophically.

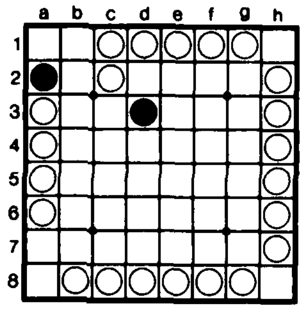


Figure 4-5: Contrived example of edge interactions. White to move.

The edge table assigns a high value to this position. The reason is simple; the evaluation of the left edge shows that white is guaranteed access to the corner at *a1*, and none of the other sides yield a certainty of black gaining a corner. However, a little search reveals that black will almost assuredly gain the remaining three sides of the board if white takes the corner. The result is that as long as white avoids the corner, the edge values are strongly in white's favor (though they should be fairly even), because of the ability to take the corner. However, should white actually take the corner, its value would plummet immediately. The source of this difficulty is that the edge table assumes a corner is always desirable as long as taking it has no bad consequences for *that* edge.

Iago handles these situations only to the extent that such sequences can be found by the normal α-β search. The lack of a mechanism that can evaluate these positions properly is one of the major remaining weaknesses in ***Iago***. One possible future resolution to this problem is to estimate dynamically the desirability of a corner by a small search like the one used for x-squares, and add information to the edge table about the values of edges when the corners are undesirable.

4.4.1.2. Internal stability

Determining which internal squares are stable is, in general, a difficult problem. However, it was possible to isolate a subclass of stable internal squares for which a feasible detection algorithm could be devised. The algorithm is based on the fact that an internal disc is stable if there is a stable disc of the same color adjacent to it on each of the four lines through the square. The algorithm makes use of two data structures: a set of unprocessed stable squares, and a boolean vector which tells for each square whether there is a stable disc on it. The algorithm is:

1. Initialize the set to be empty and all elements of the vector to be *false* (all squares unstable).
2. For each occupied corner square, mark it as stable, and add that square to the set for future processing.
3. While the set is not empty, remove an element and process it:

   For each square adjacent to the element, if it is stable (that is, there is a stable disc of the same color, or an edge of the board, adjacent to it along each of the four lines) and not so marked, then mark it stable and add it to the set.
4. The result is the difference of the number of stable internal disks for the two players (calculated from the stability vector).

The utility of this algorithm has proven to be marginal. In tight games, there are generally not a significant number of squares with this type of internal stability until it is already possible to exactly solve the position; so this algorithm has the greatest effect in games in which ***Iago*** already has an advantage. An additional problem with this algorithm is that it can miss a large number of stable internal discs. Since there is a correlation between edge stability and the stability of the near-by internal discs, working on edge stability leads to the stabilization of internal discs. Some are found by ***Iago***'s algorithm, others are not. These others can be troublesome, as they need not belong to the player with the stable edge squares. Figure 4-6 shows a position in which ***Iago***'s algorithm finds only 4 out of 12 stable internal discs. Two of them (*d7* and *e7*) belong to black, even though the stable edge and corner discs all belong to white.

The cost of executing this algorithm is decreased by not performing it during the evaluation function. By executing it one ply closer to the root node of the search tree[18], the cost is cut by a factor of 4 (the branching factor). Computing an expensive component at the terminal nodes of the search tree can result in the loss of a ply or more in the search, leading to poorer information about all components. By moving the computation of the more expensive ones out of the terminal nodes, performance on the other components is maintained at

[18]The value is then passed to all of the terminal nodes that are descendents of the node at which the calculation is performed.

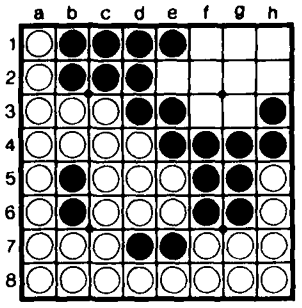


Figure 4-6: ***Iago***'s algorithm determines *b7*, *c7*, *f7*, and *g7* to be stable.
In addition, *b3*, *c3*, *b4*, *d5*, *c6*, *d6*, *d7*, and *e7* are stable.
Reversi Challenger (B-3) vs. ***Iago*** (W-60)
Santa Cruz Open Machine Othello Tournament

the higher level. In general, components should be placed so as to maximize the performance/cost ratio for the evaluation as a whole (though the actual placement in ***Iago*** is somewhat ad hoc).

4.4.2. Mobility

According to the general analysis of Othello (Section 3.2), *mobility* is the primary strategic concern during the middle game (Section 3.2.2). ***Iago***'s mobility knowledge is divided into two components: (1) *CurrentMobility* -- the number of moves currently available; and (2) *PotentialMobility* -- the potential of current board configurations to lead to future mobility.

4.4.2.1. Current Mobility

To evaluate currently available moves, ***Iago*** does the obvious, it counts them. Moves are always counted for a player just prior to when the player is to move. Thus, only the value for one player is actually computed in the evaluation function. The other player's moves are counted just before his last move in the search (usually one ply closer to the root node). It is assumed that each player has a 50% chance of taking any square that both players have access to, and 100% chance for every square to which he has sole access. A count is generated for each player by adding twice his sole access squares to the number of mutual access squares. These two counts are then non-linearly combined into a single component value. If the value for the player performing the evaluation is p, and the value for the opponent is o, then the value for the component is:

$$\text{CurrentMobility}(p, o) = \text{Truncate}[1000 \times (p - o)/(p + o + 2)] \qquad (1)$$

A strict difference between the two values is not used because having an advantage of 4 to 1 is more valuable than an advantage of 15 to 11.

Figure 4-7 traces the value of the weighted measure over the game between ***Iago*** and ***Reversi Challenger*** at the Santa Cruz Open. In this game, the transition from the middle to the end game occurs between moves 30 and 32, when the advantage in legal moves translates into domination of the edges.

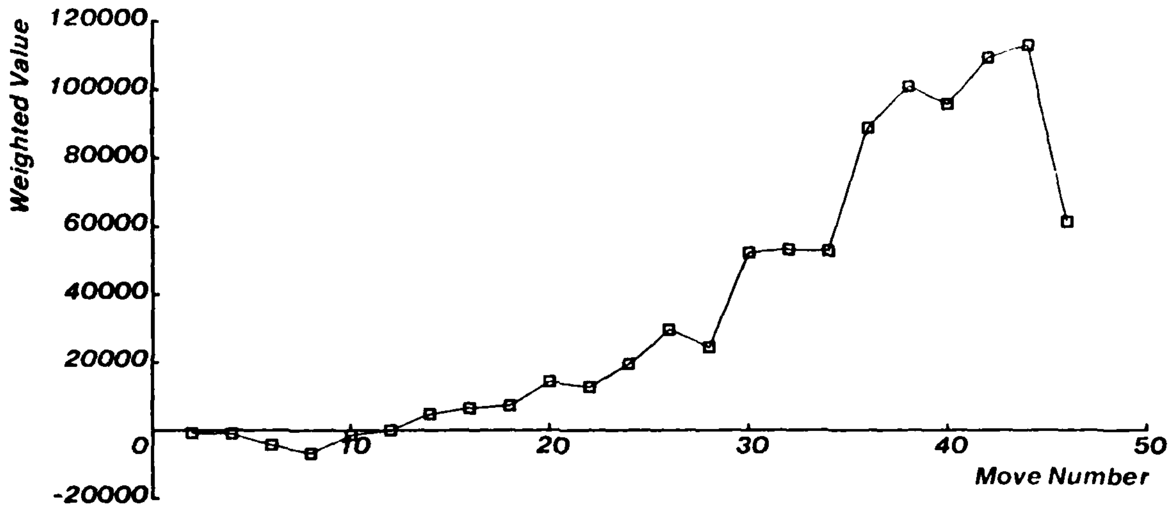
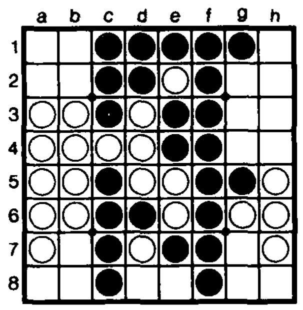
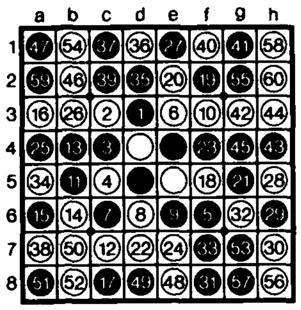


Figure 4-7: ***Iago***'s weighted legal move measure
Reversi Challenger(B-3) vs ***Iago***(W-60)
Santa Cruz Open Machine Othello Tournament

Legal move computation is one of the most costly portions of the evaluation function. Averaged over the six positions used in the evaluation of ordering heuristics (Section 4.2.1), the mean cost of counting the legal moves for one player in the evaluation function is 3.4 msecs, compared with .5 msecs for all of the other calculations (yielding a total of 7.3 msecs per evaluation when legal moves are counted for both players). By doing the computation for one player at the nodes one closer to the root node, the cost is split over 4 (the branching factor) terminal nodes, yielding a net cost of 4.75 msec per evaluation. To minimize the cost even further, legal moves are counted last, and the evaluation is terminated prematurely when either: (1) the maximum value possible for this measure would not lead to an evaluation better than the current best node; or (2) the minimum value possible for this measure would lead to a cutoff in the search tree. These conditions are checked before each square is evaluated for a possible legal move. This optimization trims another 0.8 msecs per evaluation, for a net of 3.95 msecs.

One of ***Iago***'s current weaknesses is that the legal-move measure does not distinguish between acceptable and unacceptable moves (Section 3.2.2). Figure 4-8a is a graphic example of this problem from the game between ***Iago*** and ***Aldaron*** (written by Charles Heath) at the Santa Cruz Open. At move 42, ***Iago*** has eleven moves available. However, seven of those moves result in the immediate loss of a corner. Because ***Iago*** has plenty of mobility, and the necessity of giving up a corner is over the search horizon, its evaluation is positive through move 40 (Figure 4-9). As soon as ***Iago*** realizes that it will have to play one of the bad moves, its evaluation plummets. Though ***Iago*** did win by a score of 34 to 30, ***Aldaron*** had a sure 34 to 30 victory until it unaccountably made the wrong move at move number 59[19].


(a) White to move (move 42)
7 of 11 moves are bad


(b) The game record

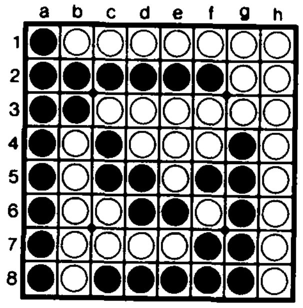

(c) The final position

Figure 4-8: Seven of ***Iago***'s eleven moves give up a corner immediately
Aldaron (B-30) vs. ***Iago*** (W-34)
Santa Cruz Open Machine Othello Tournament

The desirability of possible moves can of course be computed by searching the game tree below each move. However, because this must be done at the terminal nodes of the regular search, it is prohibitively expensive. Another alternative is to divide the squares up into a priori acceptable and unacceptable groups, as is done by ***Aldaron*** [Cerf 81b]. A third alternative, and the one that will probably be tried in ***Iago***, is to keep the current measure, and to add a second measure in which moves are eliminated if they lead to an immediate surrendering of a corner square (assuming that the corner was not already available by some other path).

4.4.2.2. Potential mobility

Iago computes three measures that combine to assess two of the three aspects of potential moves (Section 3.2.2): (1) a string of the opponents discs with (2) an empty square at one end. The three measures are:

- The number of discs that the opponent has adjacent to empty squares (*frontier discs*).
- The number of empty squares adjacent to the opponent's discs.

[19]Thus in some sense ***Iago*** won the tournament by a fluke, since this game was ***Aldaron***'s only loss. However, in an unofficial rematch, ***Iago*** beat ***Aldaron*** 40 to 24 (Figure 4-10).

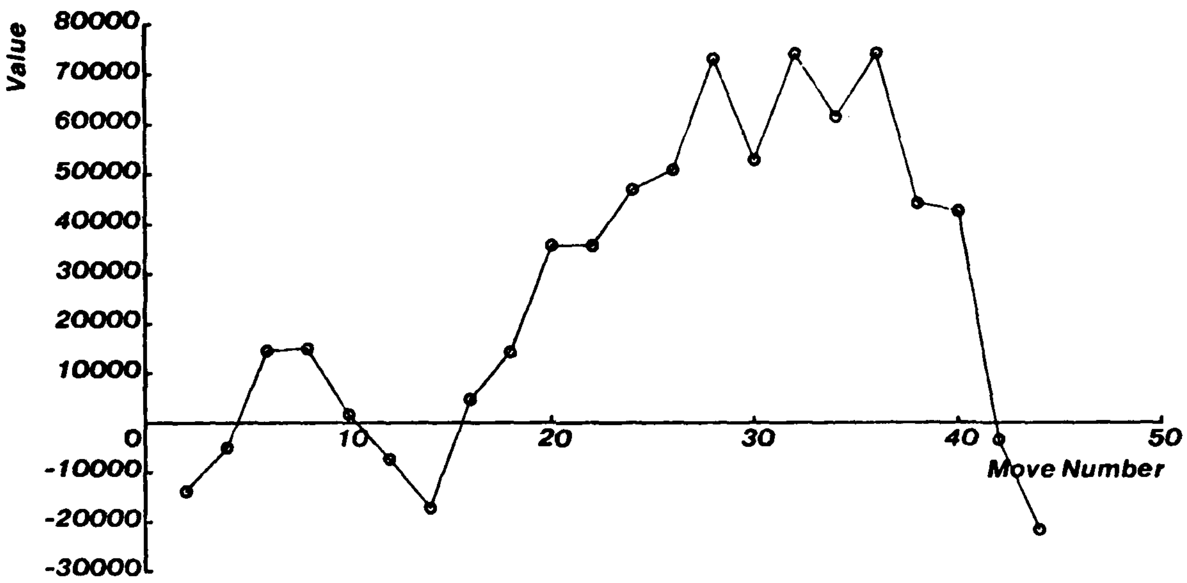
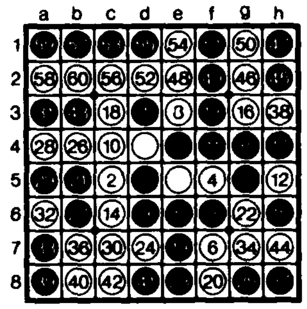


Figure 4-9: *Iago*'s evaluation on each of its moves
Aldaron (B-30) vs. *Iago* (W-34)
Santa Cruz Open Machine Othello Tournament


(a) The game record

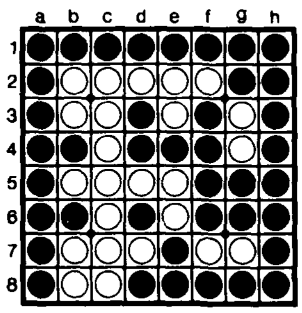

(b) The final position

Figure 4-10: ***Iago*** **(B-40) vs.** ***Aldaron*** **(W-24)**
Unofficial rematch
Santa Cruz Open Machine Othello Tournament

- The sum of the number of empty squares adjacent to each of the opponent's discs (while this is like the previous measure, empty squares that are adjacent to more than one of his discs will be counted multiply -- once for each disc).

Individually, these measures cannot capture all of the important aspects, but collectively they capture much of it. The first measure establishes one type of bound on the number of potential moves by counting the discs that can be used as a terminus of a string of discs to be flipped. The second measure establishes the complementary bound by counting the number of squares onto which a player can move. The third measure essentially repeats the second, with each blank being weighted by the number of discs to which it is adjacent. The more possible avenues there are to an empty square, the more likely it is that at least one will be converted into a legal move.

Each measure generates two values (one for each player) which are combined into a single value for each measure by Equation 1 (Section 4.4.2.1). The three values (one for each measure) are then summed to yield a single value for this component. Figure 4-11 shows each measure individually weighted, and the combined weighted value for the game at the Santa Cruz Open between *Iago* and *Reversi Challenger*.

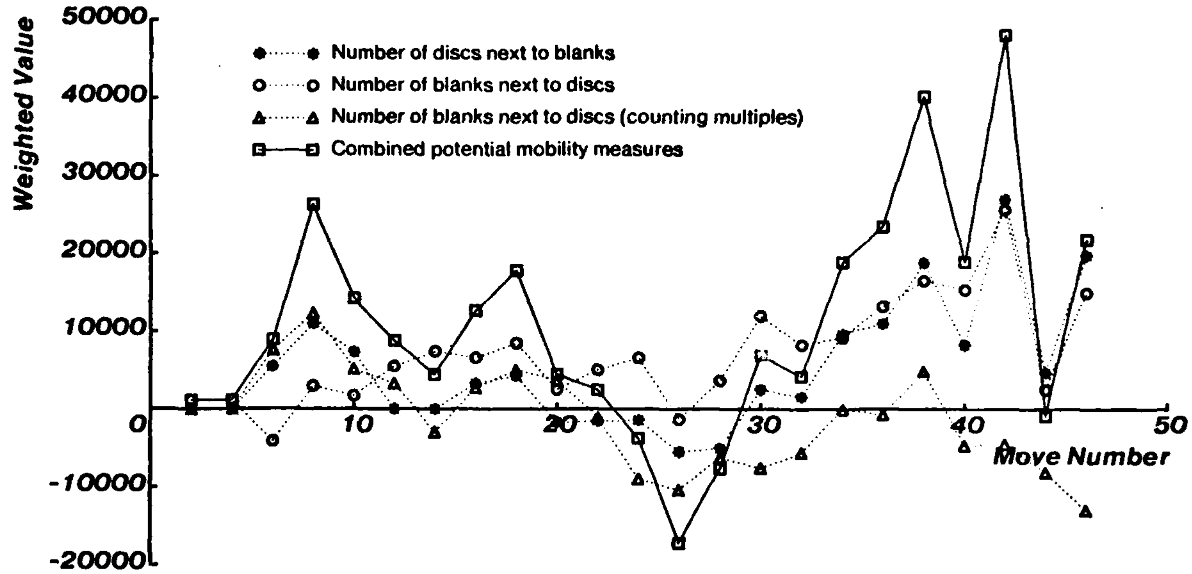


Figure 4-11: *Iago*'s potential mobility measures (for each of its moves)
Reversi Challenger (B-3) vs. *Iago* (W-60)
Santa Cruz Open Machine Othello Tournament

There is one aspect of the mobility potential of a position that is not captured by these measures: (3) the player must have a disc at the other end of the string of discs to be flipped. *Iago* currently has no mechanisms that directly address this condition. Because of the greediness of the current edge table (Section 4.4.1.1), *Iago* can frequently fulfill this condition by using an edge disc. The alternative non-greedy edge table had trouble for precisely this reason; it lacked end discs that could lead to legal moves. One possible solution is to define a set of measures based on adjacency between discs of opposite colors; for example, the number of discs the opponent has adjacent to the player's discs. This idea has not been tested in ***Iago***.

4.4.3. The opening game

In the current version of ***Iago***, the opening game is played like the middle game, with the exception of changes in the application coefficients. ***Iago*** does not have an opening book, and it is difficult to tell how much it is hurt by the lack of one. There are games, such as the one at the Santa Cruz Open between ***Iago*** and ***Aldaron*** (Figure 4-8), in which ***Iago*** fell behind in the opening game (Figure 4-9), and never quite recovered. When the colors were reversed in the unofficial rematch (Figure 4-10), forcing ***Iago*** to play a different opening, it won by a 40 to 24 score. It is unclear whether to attribute this turnaround to the color, or to the opening. During the ten games played by ***Iago*** at the Santa Cruz Open tournament (including the unofficial rematch against ***Aldaron*** and one against ***Reversi Challenger*** that was won by ***Iago*** 37 to 27), ***Iago*** played white 6 times for an average score of 41.33 to 22.33, and black 4 times for an average score of 48.25 to 15.75. When the games in which ***Iago*** played white are broken up according to the opening move, the results are: the *diagonal* opening was played 2 times for an average score of 37 to 26.5; the *perpendicular* opening was played 4 times for an average score of 43.5 to 20.25; and the *parallel* opening was not played.

The most promising approach to generating an opening book for ***Iago*** appears to be to use ***Iago*** itself to precompute choices for opening lines of play[20].

[20] We understand that ***Belle*** uses this technique to extend its chess opening book

5. Comparison of Iago with Human Play

The current version of ***Iago*** has not had the opportunity to play against championship level human competition. An attempt to arrange a match between ***Iago*** and Jonathan Cerf, the current world Othello champion, was gracefully rebuffed [Cerf 81b]:

> I understand Paul Rosenbloom is interested in arranging a match against me. Unfortunately my schedule is very full, and I'm going to see to it that it remains that way for the foreseeable future.

As an alternative to direct play, ***Iago***'s performance can be compared with that of expert players by examining ***Iago***'s analyses (at tournament time allocations, except no tournament overhead expenses are charged) of expert games. ***Iago*** processes each game, a move at a time. For each position in the game, ***Iago*** first determines the move it would make; computing and saving the value for that move,and the individual components of the evaluation. A value is then computed for the move actually played in the game, by playing the move, and then searching to a depth one less than the depth used to search for ***Iago***'s move.

This information forms the basis for three types of comparisons between ***Iago*** and the expert players: (1) the frequency with which ***Iago***'s moves are equivalent to those of the experts[21]; (2) the evaluation for ***Iago***'s choices and the experts' choices; and (3) a qualitative examination of whether the components of ***Iago***'s evaluation correlate with winning. These comparisons have been made for the two-game 1980 world championship match between Jonathan Cerf (U.S.A.) and Takuya Mimura (Japan). The first game has already been presented in Figure 3-9[22]. The second game in the match can be seen in Figure 5-1[23]. Both games were won by Cerf. Figures 5-2 and 5-3 show the moves that ***Iago*** would have made at each point in the two games[24].

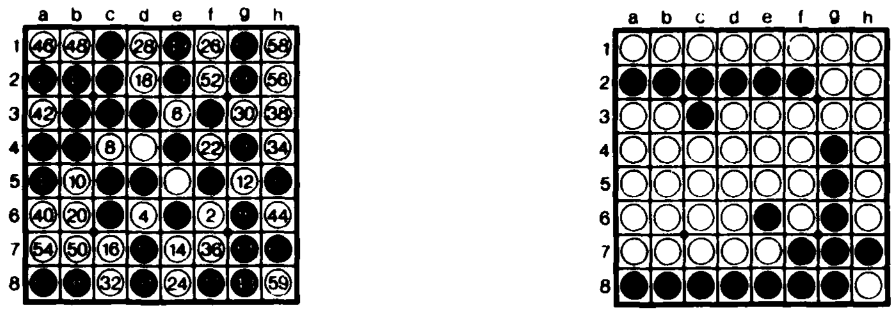
Table 5-1 summarizes the first two comparisons. Moves 46 to 60 are listed separately because ***Iago*** is the expert in that region -- it solved those positions during the analysis. Moves 39 and 41 by Cerf in the first game were eliminated from the analysis because ***Iago*** radically misestimates their values. The board position just before move 39-*h2* can be seen in Figure 5-4a. Black's play at *h2*, allows white to gain the corner at *h1* (by first playing at *h5*). As long as white can avoid playing at *h1*, the right side is in his favor, and the top side is acceptable. Once white plays at *h1* though, the right side is still in his favor, but the top side is highly

[21]Two moves are *equivalent* but not *identical* only when the choice is between symmetric moves (such as the first move of the game), or the two moves would result in the same final score (applies in those end-game positions that are solvable by ***Iago***).

[22]This game has been analyzed by Cerf (1981a).

[23]This game has been analyzed by Sullivan (1981).

[24]Because ***Iago*** does not always make the same choice as was actually played in the game, it may suggest the same move at more than one move number, leading to repetitions in these records.



(a) The game record **(b)** The final position

Figure 5-1: Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

1-B-*d3*	11-B-*g6*	21-B-*d7*	31-B-*c7*	41-B-*b7*	51-B-*a1*
2-W-*d6*	12-W-*h5*	22-W-*f1*	32-W-*b6*	42-W-*a7*	52-W-*g7*
3-B-*c3*	13-B-*c4*	23-B-*d7*	33-B-*a4*	43-B-*g6*	53-B-*a1*
4-W-*f4*	14-W-*c3*	24-W-*d7*	34-W-*b5*	44-W-*b2*	54-W-*h7*
5-B-*e3*	15-B-*g3*	25-B-*h4*	35-B-*a6*	45-B-*f7*	55-B-*a8*
6-W-*c6*	16-W-*c2*	26-W-*d7*	36-W-*b5*	46-W-*g7*	56-B-*b7*
7-B-*f3*	17-B-*f2*	27-B-*c7*	37-B-*b5*	47-B-*e8*	57-B-*c8*
8-W-*g6*	18-W-*b4*	28-W-*e1*	38-W-*d8*	48-W-*f8*	58-W-*b8*
9-B-*c4*	19-B-*e1*	29-B-*c1*	39-B-*a7*	49-B-*c8*	59-B-*g8*
10-W-*f3*	20-W-*f1*	30-W-*b1*	40-W-*a2*	50-W-*h1*	60-W-*h8*

Figure 5-2: The moves that ***Iago*** selects when faced with the game positions
First game of the 1980 World Championship match
Cerf (B-44) vs. Mimura (W-20)

1-B-*d3*	11-B-*b6*	21-B-*c3*	31-B-*d8*	41-B-*f2*	51-B-*g7*
2-W-*d6*	12-W-*g4*	22-W-*f4*	32-W-*c8*	42-W-*a3*	52-W-*f2*
3-B-*e6*	13-B-*a4*	23-B-*a5*	33-B-*b3*	43-B-*f2*	53-B-*g1*
4-W-*f4*	14-W-*e7*	24-W *e8*	34-W-*g8*	44-W-*h6*	54-W-*a7*
5-B-*c5*	15-B-*a4*	25-B-*a5*	35-B-*f7*	45-B-*b2*	55-B-*g1*
6-W-*f4*	16-W-*c3*	26-W-*c2*	36-W-*f7*	46-W-*a1*	56-W-*h2*
7-B-*d3*	17-B-*h5*	27-B-*c2*	37-B-*a4*	47-B-*a2*	57-B-*g1*
8-W-*g5*	18-W-*d2*	28-W-*f2*	38-W-*h3*	48-W-*f2*	58-W-*h1*
9-B-*c6*	19-B-*c3*	29-B-*c2*	39-B-*g8*	49-B-*e1*	59-W-*h7*
10-W-*b5*	20-W-*f3*	30-W-*f2*	40-W-*a6*	50-W-*b7*	60-B-*h7*

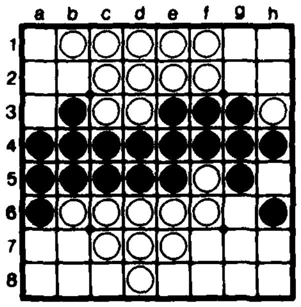
Figure 5-3: The moves that ***Iago*** selects when faced with the game positions
Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

negative. White does eventually have to play at *h1*, but it is beyond the horizon of ***Iago***'s search. Similarly, at move 41 (Figure 5-4b), black's play at *a3* allows white to push the *h1* move over the search horizon. This is exactly the sort of problem that can be caused by the independence assumption built into the table of side values.

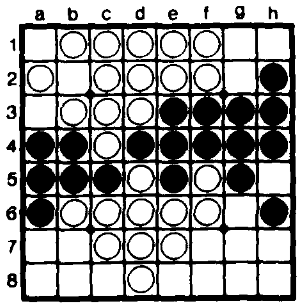
Game Number	Player	Score	Equivalent Move Fraction			Mean Difference	
			1 - 45	46 - 60	1 - 60	Evaluation 1 - 45	Disc Count 46 - 60
1	Cerf*	44	.52	.88	.62	4214	0.7
1	Mimura	20	.55	.86	.62	7189	4.0
2	Mimura	21	.43	.86	.53	4527	1.4
2	Cerf	43	.50	.88	.60	2227	0.5
Subtotal	Cerf*	87	.51	.88	.61	3197	0.6
Subtotal	Mimura	41	.49	.86	.58	5828	2.7
Total		128	.50	.87	.59	4543	1.6

* Cerf's moves 39 and 41 were eliminated from game 1 prior to this analysis, because ***Iago*** totally misjudges them (see text for discussion).

Table 5-1: ***Iago***'s analysis of the 1980 world championship match (Moves 2-45)
Jonathan Cerf (U.S.A.) vs. Takuya Mimura (Japan)



(a) Before black's 39-*h2*



(b) Before black's 41-*a3*

Figure 5-4: Two positions in which ***Iago*** misestimates the value of a move
Cerf (B-44) vs. Mimura (W-20)
First game of the 1980 World Championship match

5.1. Equivalent move fraction

Iago duplicates the effect of 50 percent of the moves made by the two experts while using its inexact evaluation (moves 1-45), and 59 percent of their moves over the entire game. Interpretation of these percentages is complicated by the possibility that ***Iago*** is better than the experts. If so, a high degree of match is a reflection of the expert's capability, not of ***Iago***'s. It would be interesting to compare this with a typical percentage of agreement between two experts, but that information is not available.

The values do compare favorably with the 38 percent match against expert moves (identical moves only) reported by Samuel for his checker program [Samuel 67]. Samuel achieved a match against 64 percent of the experts' moves when both of the top two choices were included. A corresponding value cannot be generated for ***Iago*** because its search procedure is only guaranteed to have the best move ordered correctly; all of the other alternatives may be out of order.

Another question that can be asked about this data is how well the fraction correlates with the outcome of the game. In the first game, both players agreed with ***Iago*** 62 percent of the time. The lack of a difference between the percentages for the two players, combined with the lopsided outcome of the game (a 24 disc victory by Cerf), yields a poor correlation between the percentages and the outcome. However, it turns out that if Mimura had played 46-*g7* instead of 46-*b2*, he would have been assured of losing by no more than 2 discs -- a result consistent with the percentages. In the second game, ***Iago*** agrees with 60 percent of Cerf's moves and 53 percent of Mimura's moves -- a result consistent with a 22 disc victory by Cerf.

5.2. Positional value

The last two columns in Table 5-1 contain the mean differences between the values of the moves selected by ***Iago*** and those actually made by the experts. During the early portions of the game (moves 1 to 45), both values are computed by ***Iago***'s evaluation function. Since ***Iago***'s evaluation will always assign a higher value to its own selection than to the move made by the expert (unless the search is aborted before the expert's move is found), the differences are always positive. By comparing these differences for the two experts, conclusions can be drawn about the closeness of ***Iago***'s style of play to that of the experts. In both games, the mean difference is smaller for Cerf than for Mimura. The same is apparent when the differences are looked at in more detail. Figure 5-5 shows the value for each position in which Mimura was to move (in the second game). The two curves represent the values for ***Iago***'s selection and Mimura's selection. Figure 5-6 shows the equivalent curves for positions in which Cerf is to play.

These curves reinforce the conclusion that ***Iago***'s evaluation is closer in effect to Cerf's style of play, than to Mimura's. They also reveal that Mimura probably erred late in this game as well. At move 39, ***Iago*** assigns a

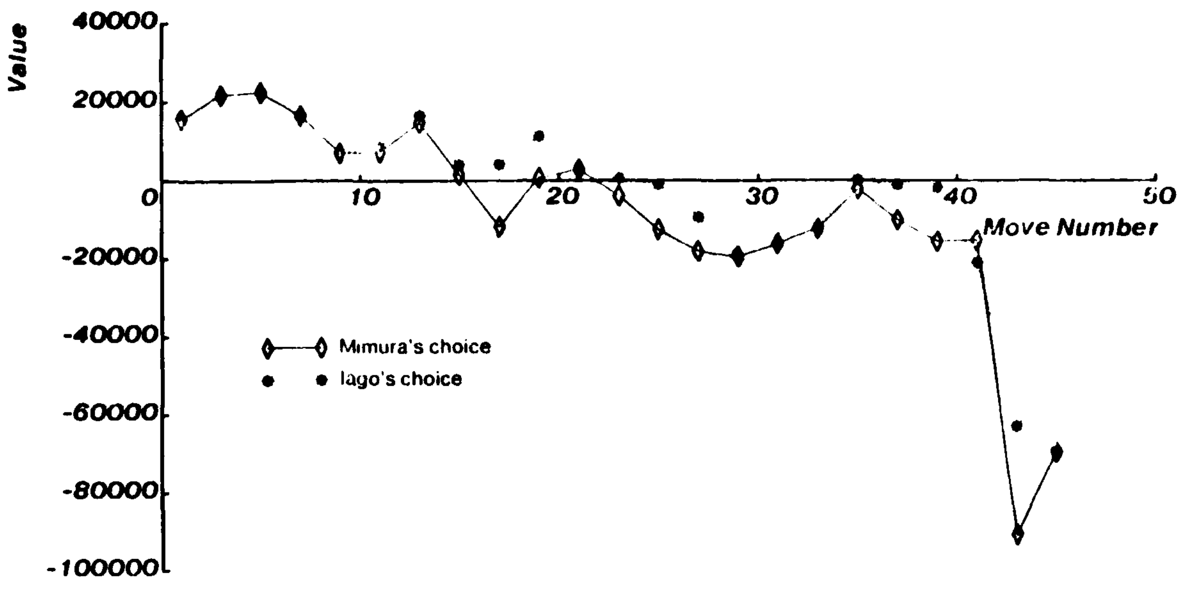


Figure 5-5: ***Iago***'s evaluation of the moves made by Mimura, and its own choices
Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

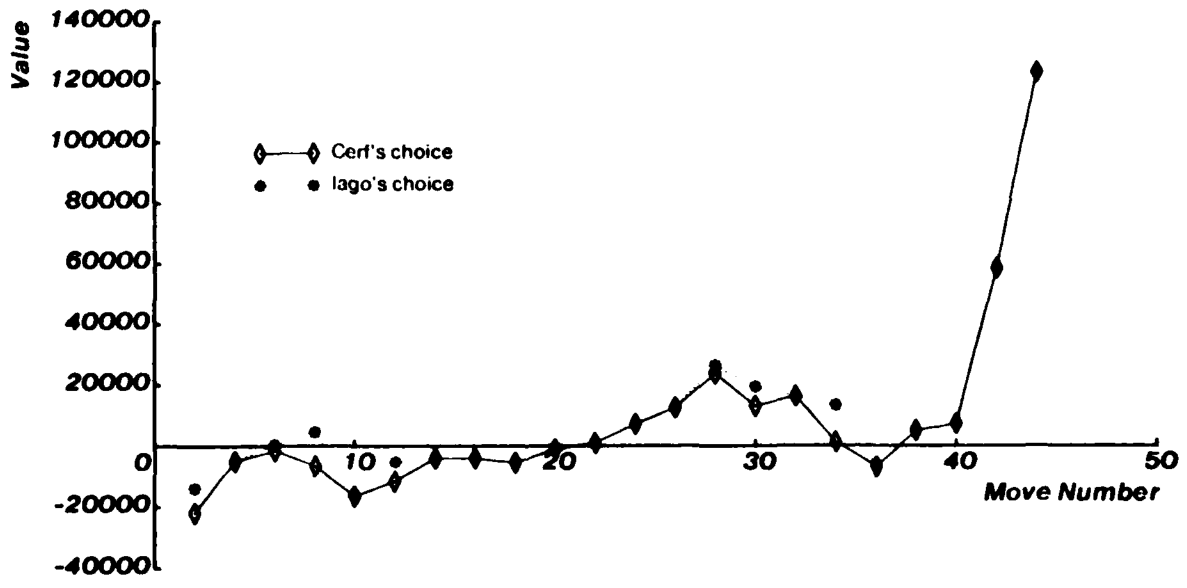
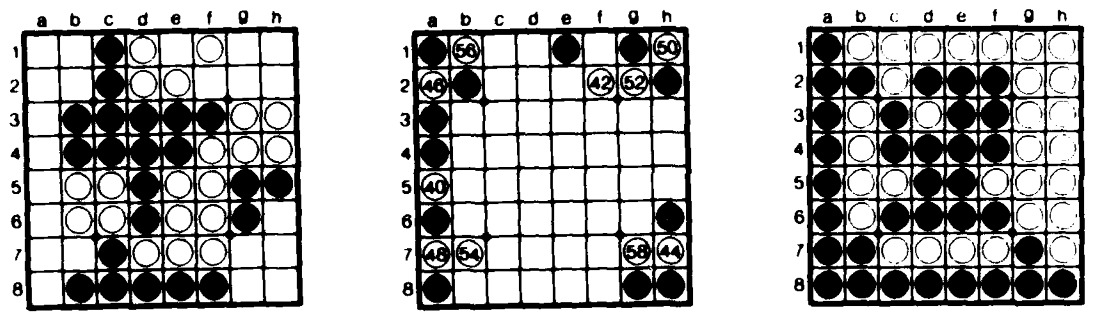


Figure 5-6: ***Iago***'s evaluation of the moves made by Cerf, and its own choices
Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

value of -1820 to Mimura's position. This is a very small value for this point in the game (against ***Reversi Challenger***, ***Iago*** assigned a value of -627731 to ***Reversi Challenger***'s position at move 39); certainly it should not be enough of a problem to cause a loss by 22 discs. To investigate this, ***Iago*** was set up to play against itself with a constant time allocation of 2 minutes per move, starting with the board position immediately following move 38-h3 by Cerf (Figure 5-7a). The game continuation can be seen in Figure 5-7b: black (Mimura's color) won by 4 discs (Figure 5-7c). ***Iago***'s evaluation at move 39 is consistent with this outcome.



(a) The board before move 39 **(b)** The game record from move 39 **(c)** The final position

Figure 5-7: ***Iago***'s continuation of the second game of the 1980 world championship match
The first 38 moves were take from the game record (Figure 5-1a)
Black wins 34 to 30

During the latter portion of the game (moves 46 to 60), end-game searches are performed to evaluate the moves selected. ***Iago***'s moves were optimal, so the differences reflect the mean number of discs lost by the experts as a result of their end-game moves.

5.3. Componential analysis

Figure 5-8 shows the weighted edge value, determined by search, at each point in the game. It shows the same general character as the one in Figure 4-3; a long stretch in which only minor advantages are gained, followed by an abrupt skyrocketing of the value. This skyrocketing occurs after the bad moves by Mimura. Figure 5-9 shows the legal-move and potential-mobility measures for the game. Mimura's early lead (Figure 5-5) was due to his advantage in potential mobility, but he was unable to convert it into a legal-move advantage. Neither side ever achieved a large mobility advantage, except at the very end when mobility was traded off for edge advantage. One interesting aspect of these two curves is that an advantage in legal moves is always preceded by a peak in the potential mobility curve. This is a good example of how the potential mobility measures can serve as a predictor of future legal moves.

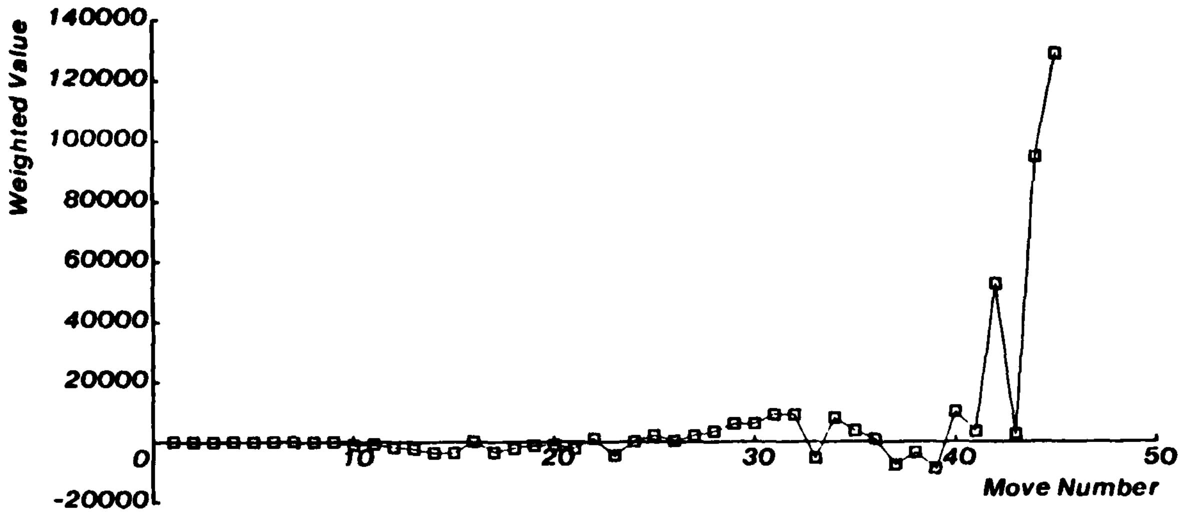


Figure 5-8: The weighted edge value (from Cerf's point of view)
Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

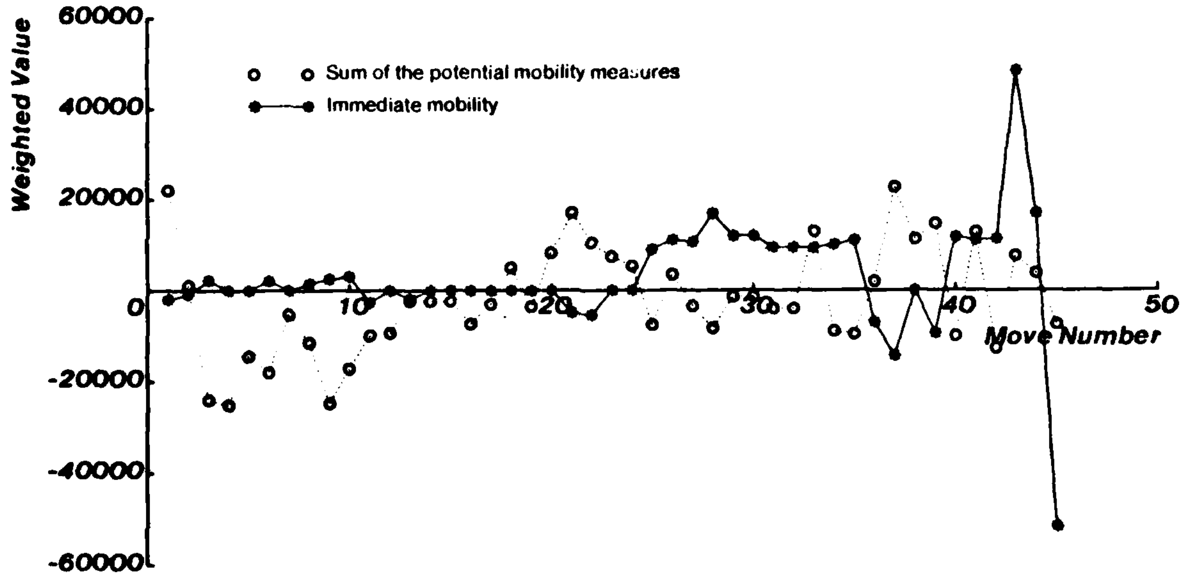


Figure 5-9: The weighted legal-move and combined potential-mobility measures (from Cerf's point of view)
Second game of the 1980 World Championship match
Mimura (B-21) vs. Cerf (W-43)

6. Conclusion

Modern artificial intelligence programs have been developed for a few games (chess, checkers, backgammon, and kalah (see Slagle (1971))). Now ***Iago*** adds a successful example for the game of Othello. All of the major mechanisms that have emerged in game playing programs are used in some form. So ***Iago***'s world-championship level of play may be taken as evidence for the basic adequacy of the existing class of mechanisms developed in artif[illegible] intelligence.

Development of this application involved: (1) a task analysis, yielding knowledge about the game and a control structure for its application; (2) implementation of the knowledge and structure through state-of-the-art techniques; and (3) an analysis of the performance of the resulting system.

Othello has been analyzed into a pair of major strategic concepts (stable territory and mobility), each decomposable into sub-concepts. Combined with the α-β search algorithm, iterative deepening, and move ordering, these concepts form the basis of most of ***Iago***. Additional aspects of ***Iago*** handle end-game searches, time allocation, and the efficiency of both the algorithms and the implementation.

Iago has been evaluated by: (1) direct play against other programs -- resulting in a 10-0 record; (2) an analysis of the effectiveness of its mechanisms as implementations of the underlying Othello concepts; and (3) comparing its analyses of expert games with the experts' play -- showing that ***Iago*** is able to approximate their level of performance, while avoiding some of their major errors.

Acknowledgement

Many people have contributed substantially to this effort. The first version of ***Iago*** was written while visiting the LNR group at the University of California, San Diego. They provided invaluable encouragement and computer time. More recently, many of the ideas came from discussions with Charles Leiserson and Bruce Ladendorf. Bruce also programmed the precomputation algorithm for the edge table. Allen Newell has provided helpful comments over many iterations of this article. Hans Berliner was always available for sage advice on game playing. Peter Frey organized both tournaments which provided invaluable experience. Kate Rosenbloom ran ***Iago*** at the Santa Cruz tournament, and Bruce Ladendorf assisted with running ***Iago*** at the Northwestern tournament. David Lamb and Craig Everhart provided support for (and assistance with) ***Sail***. Helpful comments have also been received from Glenn Iba, John Laird, and Andrew Palay. Finally, thanks have to go to the whole CMU computer science community for providing an encouraging environment for this work and for putting up with the demand that ***Iago*** has occasionally placed on the computer resources.

References

[Albion 80] Albion, H.
Japanese openings.
Othello Quarterly II(2):3-5, Summer, 1980.

[Berliner 80] Berliner, H. J.
Backgammon computer program beats world champion.
Artificial Intelligence 14(2):205-220, September, 1980.

[Cerf 81a] Cerf, J.
Cerf vs. Mimura.
Othello Quarterly II(4):16-21, Winter, 1980/81.

[Cerf 81b] Cerf, J.
Machine vs. machine.
Othello Quarterly III(1):12-16, Spring, 1981.

[Cichelli 73] Cichelli, R. J.
Research progress report in computer chess.
ACM Sigart Newsletter (41):32-36, June, 1973.

[Frey 80a] Frey, P. W.
Simulating human decision-making on a personal computer.
BYTE 5(7):56-72, July, 1980.

[Frey 80b] Frey, P. W.
Machine Othello.
Personal Computing :89-90, July, 1980.

[Frey 81a] Frey, P. W.
The Santa Cruz open Othello tournament for computers.
BYTE 6(7):26-37, July, 1981.

[Frey 81b] Frey, P. W.
Personal communication.
1981.

[Gillogly 78] Gillogly, J. J.
Performance analysis of the Technology chess program.
PhD thesis, Carnegie-Mellon University, 1978.

[Hasegawa & Brady 77]
Hasegawa, G., and Brady, M.
How to Win at Othello.
Jove Publications, New York, 1977.

[Jacobs & Jacobs 79]
Jacobs, C., and Jacobs, E.
Unbalancing your opponent.
Othello Quarterly I(1):3-5, Spring, 1979.

[Maggs 79] Maggs, P. B.
Programming strategies in the game of Reversi.
BYTE 4(11):66-79, November, 1979.

[McCarthy 81] McCarthy, J.
Personal communication.
1981.

[Nilsson 71] Nilsson, N. J.
Problem-Solving Methods in Artificial Intelligence.
McGraw-Hill, New York, 1971.

[Personal 80] Personal Computing.
Background and origins of Othello.
Personal Computing :87-88, July, 1980.

[Phillips 80] Phillips, R.
Writing an Othello program.
Othello Quarterly I(3):7-12, Winter, 1979/80.

[Prentice 81] Prentice, C.
Report from London.
Othello Quarterly II(4):10-12, Winter, 1980/81.

[Richards 81] Richards, R.
The revised USOA rating system.
Othello Quarterly III(1):18-23, Spring, 1981.

[Samuel 63] Samuel, A. L.
Some studies in machine learning using the game of checkers.
In E. A. Feigenbaum & J. Feldman (editors), *Computers and Thought*, pages 71-105. McGraw-Hill, New York, 1963.

[Samuel 67] Samuel, A. L.
Some studies in machine learning using the game of checkers, II - Recent progress.
IBM Journal of Research and Development 11(6):601-617, November, 1967.

[Slagle 71] Slagle, J. R.
Artificial Intelligence: The Heuristic Programming Approach.
McGraw-Hill, New York, 1971, pages 26-28.

[Slagle & Dixon 69]
Slagle, J. R. and Dixon, J. K.
Experiments with some programs that search game trees.
Journal of the Association for Computing Machinery 16(2):189-207, April, 1969.

[Slate & Atkin 77]
Slate, D. J., and Atkin, L. R.
CHESS 4.5 -- The Northwestern University chess program.
In P. W. Frey (editor), *Chess Skill in Man and Machine*, pages 82-118. Springer-Verlag New York, 1977.

[Stringham 80] Stringham, G.
Fundamental Othello misconceptions: Disc-counting strategies.
Othello Quarterly II(3):3-7, Fall, 1980.

[Sullivan 81] Sullivan, G.
Playing defensively.
Othello Quarterly III(1):24-31, Spring, 1981.

[Sullivan & Richards 81]
Sullivan, G., and Richards, R.
Glossary of basic Othello terms.
Othello Quarterly II(4):8-9, Winter, 1980/81.