Towards Perfect Play of Scrabble®

ISBN 905278 351 9
Universitaire Pers Maastricht

Towards Perfect Play of Scrabble®

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit Maastricht,
op gezag van de Rector Magnificus, Prof. dr. A.C. Nieuwenhuijzen Kruseman
volgens het besluit van het College van Decanen,
in het openbaar te verdedigen
op vrijdag 5 juli 2002 om 12.00 uur

door

Brian Sheppard

UM
UNIVERSITAIRE PERS MAASTRICHT

Promotores:
Prof. dr. H.J. van den Herik
Prof. dr. J. Schaeffer (University of Alberta)

Beoordelingscommissie:
Prof. dr. A. J. van Zanten (voorzitter)
Dr. A.P.J. van den Bosch (Katholieke Universiteit Brabant)
Prof. dr. A. de Bruin (Erasmus Universiteit Rotterdam)
Prof. ir. L.A.A.M. Coolen
Prof. dr. H. Visser


Dissertation Series No. 2002-10
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Printed by Datawyse, Maastricht, The Netherlands.

Contents

List of Figures

Preface

Computer Scrabble programs have achieved a level of performance that exceeds that of the strongest human players. MAVEN was the first program to demonstrate this against human opposition. Scrabble is a game of imperfect information with a large branching factor. The techniques successfully applied in two-player games such as chess do not work here. MAVEN combines a selective move generator, simulations of likely game scenarios, and the B* algorithm to produce a Scrabble-playing program of super-human skill.

I became involved in computer Scrabble because of a personal fascination with creating high-performance game-playing engines. That fascination alone sustained the project for many years. Truly, I never conceived of this work becoming a doctoral dissertation. Credit for believing that this work was worthy must go to Professor Jaap van den Herik and Professor Jonathan Schaeffer. I was content to continue my career in professional programming, without a care for publishing anything about MAVEN. However, Jaap and Jonathan would have none of that. Jonathan firmly stated that MAVEN merited a serious academic paper. While I did not mind writing a paper, I had no patience for finding a publisher. Well, Jonathan went out and found a publisher, so I had to write a paper. Jaap then said that an expanded version would make a fine dissertation, and would I mind writing one? Jaap removed all obstacles and objections to completing this thesis, until I had no alternative.

I am grateful to my family, and especially to my wife, who supports my efforts unfailingly.

I thank the many Scrabble experts who contributed to my understanding of the domain. I must single out Joel Wapnick, Joe Edley, and Nick Ballard, who have taught me a great deal. I have an enduring admiration for your artistry and an enduring affection for you personally.

I thank Hasbro, Inc., for their support of Scrabble in general and MAVEN in particular.

Chapter 1 – Introduction

Scrabble is a board game in which players build words with tiles containing letters of varying point values. Players from across the world enjoy Scrabble as a pleasant family board game for two to four players. In this thesis, we consider only its tournament variant, which is a two-player game. The game's rules are in Appendix A.

1.1 Why Scrabble?

Scrabble is a good platform for testing Artificial Intelligence (AI) techniques. The game has an active competitive tournament scene, with numerous national and international Scrabble associations and a biennial world championship for English-language players. There are many levels of skill. Creating a program that simply plays legal moves is a significant challenge. Once the program plays legal moves, you find that it is too weak to challenge top humans, and nothing obvious will address the shortcomings.

Even though Scrabble is one of the most popular games in the world, with millions of game sets sold annually, little has been published in the computer literature about the game. In part, this is because the game-AI tradition focuses on deterministic games. Chess, checkers, reversi, awari, hex, go, renju, amazons, Pente, etc., dominate the game AI literature. Some of these games did not even *exist* when serious investigations into Scrabble began in the mid-1970s, yet the literature still emphasizes deterministic games. In a small way, this thesis will redress the balance by publishing investigations into a game having a significant random component.

Similarly, the game-AI literature focuses on perfect-information games. The games listed above are all perfect-information games. Additionally, there is a computer literature for backgammon, which is a stochastic game of perfect information. We are starting to see some literature about games with private information, like poker and bridge, and this thesis will add to that trend.

1.2 Some Characteristics of Scrabble

Scrabble has a large state space. The state space is *much* bigger than chess, and is even bigger than Go. The number of states is perhaps 2^{1000}. The number of legal moves is large, too. An average game position has about 800 moves, and positions can have over 8000 moves.

However, these imposing parameters of the state space are perhaps misleading, since expert Scrabble play is not predominantly concerned with exhaustively searching the state space. In part, this is because Scrabble has a random component, as the players draw tiles blindly. The random component gives the state space a high variance, so that the way that a move turns out depends on many imponderable factors. Because you cannot calculate the outcome of a move in most cases, in Scrabble evaluation takes precedence over search.

Move generation in Scrabble is a challenging task. To figure a rough order of magnitude for the task, on each turn you can place any subset of your 7 tiles in any order either horizontally or vertically starting at any of 225 squares of the board. A rough order of magnitude calculation shows 2 * 225 * 7! = 2,268,000 potential words to check for

legality. Checking a move for legality involves looking up all words created or modified by a play in a dictionary containing about 150,000 words of from two through 15 letters.

On a modern computer you could use brute-force calculations for this computation, but even with today's computers the program would be too slow for certain purposes. With the computers of the mid-1970s, when serious investigation of computer Scrabble first began, a brute-force approach was impossible. It was essential to discover ways to systematically prune the decision process to generate all legal moves with a minimum number of blind alleys.

Move generation is, at least, feasible for computers. Human opponents, in contrast, have great difficulty with move generation. If you compare Scrabble with chess, you would find that human Scrabble champions have a best-play percentage of about 50%, whereas chess champions are close to 99%.[1] Simply finding legal moves is a difficult cognitive challenge in Scrabble, whereas it is not a challenge in chess.

Additionally, Scrabble poses challenges beyond move generation. For example, consider a computer program that selects the highest scoring move on each turn. A human master would brush aside such a program without difficulty, winning about 75% of all games, with a spread of about 50 points per game.

Therefore, move evaluation is a crucial skill. Moreover, computer programs might be at a disadvantage, since a human's ability to integrate a wealth of experience should be difficult to reproduce in software.

Nevertheless, there is reason for hope that programs can master move evaluation. For one thing, human experts often do not agree on the merits of moves. Of course, some moves are clearly best and every player (even computer programs) would agree. However, we find that when a decision is close, there are bound to be human experts on all sides.

This suggests that even if human experts had perfect move generation, there would still be differences in skill attributable to evaluation.

1.3 Problem Statement

Is it possible to build a computer program that outplays all human Scrabble experts? Taking the extreme case, suppose that the best human expert specifically prepared for a match by playing training games against the program, so that there is no possibility that the human will lose because he is unfamiliar with the program. We desire that the program nevertheless retain an edge.

MAVEN is a computer program designed to answer that question. At the time of MAVEN's debut, it was laughable to even ask the question. Human experts were familiar with two

[1] For instance, Karpov and Kasparov have played 143 games in championship matches, with 40 decisive games. At roughly 80 moves per game, the rate of decisive (i.e., losing) moves is about 0.0035. This calculation is vulnerable to the possibility that the game-theoretic value of some games may have changed multiple times, with compensating errors resulting in a draw. The reader may judge for himself the impact of this factor. Regardless, it is clear that the act of move generation is not an impediment to chess players.

specific computer programs. Neither one played a solid game. The programs had several basic flaws.

1) They were slow, and could not complete a game in tournament time.
2) They did not have the full list of legal words.
3) They applied weak strategies.

Thus, the first question leads to three others. Can a computer program compete in real time? Can a program play using the full list of legal moves? Can a program select moves using a strategy having clear benefits over the highest-point strategy?

Even if the program passed these hurdles, there remain others. Human masters can adapt their play to trick programs. For example, there are specific techniques that humans apply when they need to come back from a deficit. In addition, they have a set of techniques that they apply to hold onto a lead. To take an example from a different domain, early expert backgammon programs were vulnerable against players that deliberately adopted specific tactics that created positions that the programs handled badly [1]. The same is true in chess; it is well-known that David Levy adopted successful anti-computer tactics in his famous matches against computers [2].

4) Can programs play correctly when holding a lead?
5) Can programs play correctly when trying to come from behind?
6) Can programs succeed despite anti-computer strategies?

Going further, is it possible to build a computer program that plays perfectly? Given that Scrabble is a game of imperfect information, is it even possible to define perfection? The ideal strategy implied by classical game theory would be a mixed strategy, where even if the payoff matrix could be calculated (which it cannot) the matrix would be so large that it is unclear whether the normal methods [3] could find the solution in a reasonable time.

Suppose that we relax the requirements a little, and ask about "practical" perfection. Is it possible to build a program that always chooses the move that maximizes winning percentage, given reasonable limitations on the opponent's ability to infer what tiles we keep after our move? You can use an "adversarial" test to measure practical perfection. If you declare that your program played a perfect game, and no one can prove to an objective third party that you are wrong, then your program is practically perfect.

If we back off practical perfection a little, then we can wonder about "asymptotic" practical perfection. Your program has this property if it will achieve practical perfection running on computers that are sufficiently fast. Microcomputer chess programs, which attained this property 25 years ago, demonstrate the value of achieving asymptotic practical perfection. Now, thanks in part to a 4000-fold increase in CPU power, these programs are knocking on the Champion's door [4]. Can Scrabble programs be asymptotically practically perfect?

1.4 Research Questions

1.4.1 Domain Analysis

What aspects of the domain are essential to strong play? We have already mentioned move generation and evaluation. Move generation is a challenging yet purely technical problem, so the crucial research question is how to evaluate. Is there an established positional theory to guide move selection?

What techniques will human opponents wield against the program? Must we develop strategies to avoid specific types of positions?

1.4.2 Move Generation

What is the best approach for generating and scoring moves? There are several reasonable approaches for generating moves, with different strengths. Which ones stand out, and under what conditions?

Besides the algorithm used, there is a software-engineering question. How do you create a move generator that is useful in different contexts? Some classic software design patterns apply.

1.4.3 Move Selection

What algorithms and heuristics sort moves in order of quality? Obviously, the score of moves is important. What other factors contribute to the decision? What computations correspond to concepts in the positional theory of the game?

Given a system of evaluation, how should we tune its parameters? We expect that any evaluation system for Scrabble will involve many parameters. We would prefer not to tune them by hand, but if we use an automated system then what metrics should the tuning process maximize? Besides the evaluation system, there is an implementation question. How can you evaluate all of the moves without adversely affecting the program's speed?

1.4.4 Competitive Proof

Can a program compete with humans in real time with competitive success? What metrics can we use to gauge the difference between human and computer play? Can we accumulate sufficient evidence to claim superiority?

1.5 Overview of Thesis

Chapter 2 describes the game of Scrabble, concentrating on the nature of the game and the skills needed to perform at a high level.

Chapter 3 sketches a MAVEN-centric view of the history of computer Scrabble. The computer science literature has little information about computer Scrabble.

Chapter 4 covers the purely algorithmic problem of move generation. In many games, it is trivial to generate moves, but in Scrabble it is difficult. Moreover, if you want to play

Scrabble at a high level, then move generation must be exhaustive and fast. Programmers have devised ingenious algorithms and data structures for meeting these requirements.

Chapters 5 and 6 cover the evaluation of moves. Scrabble stands in contrast to chess, in which a highly developed positional theory was available to computer researchers from the beginning. The opposite was true in Scrabble: computer investigations have disproved much of human expert theory. Computer Scrabble researchers needed to develop a positional theory largely from scratch. Chapter 5 shows how to evaluate changes to the rack of tiles. Chapter 6 discusses how to evaluate changes to the board.

Chapters 7, 8, and 9 show how the game evolves in stages. The first stage is the Early Game, characterized in Chapter 7. The final stage of the game is the Endgame, which Chapter 8 covers in detail. Chapter 9 deals with the Pre-Endgame, which is the transition between the two. We will tackle the Pre-Endgame last because it has aspects of both the Early Game and Endgame.

Chapter 10 describes Simulation, a technique that revolutionized computer Scrabble. The technique goes by the name *rollout* in backgammon research, and by the generic terms *stochastic lookahead* and *Monte Carlo search*. In addition to Scrabble and backgammon, the technique applies to bridge and poker,[2] so it is widely applicable. The application to Scrabble is successful, so it is worth studying closely.

Chapter 11 presents MAVEN's competitive results. MAVEN has played in three tournaments and three matches. These results validate the quality of MAVEN's implementation.

Chapter 12 describes opportunities for further investigation. While research on MAVEN has been a resounding success, there remain areas where further investigations may be fruitful.

Chapter 13 summarizes research results.

Appendix A gives the rules of the game.

Appendix B presents four annotated games. The material illustrates the caliber of strong human players, which never ceases to amaze. The annotations also illustrate the power of simulation, and the occasional shortcomings of MAVEN's implementation of simulation.

[2] And there might be some practical applications, too.

Chapter 2 – The Game of Scrabble

The goal in Scrabble is to create words, using the same constraints as in crossword puzzles. In this chapter we refrain from providing a full description of the game. We assume that any reader is familiar with Scrabble to an extent and that this superficial knowledge of Scrabble rules is sufficient to understand the essence of this chapter. Full rules are in Appendix A.

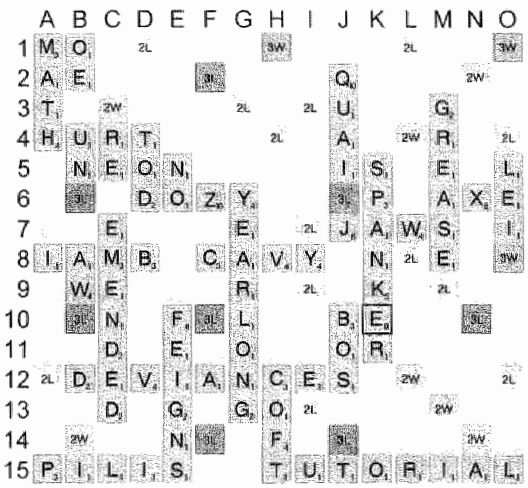
Section 2.1 describes the notations and conventions that describe moves and positions.

Section 2.2 gives fundamental metrics and statistics of the game, focusing on how to score points.

Section 2.3 describes how humans play the game. How humans cope with cognitive limitations is an interesting psychological question, and yields insight into the nature of potential computer advantages and disadvantages.

2.1 Basic Notation

This thesis makes extensive use of position diagrams like Position 2-1.



Position 2-1 Example Position

The board's premium squares are labeled 2L, 3L, 2W, and 3W to indicate double and triple premiums for letters and words. All other squares are non-premium squares.

The columns are labeled A through O, and the rows are labeled 1 through 15. This establishes a coordinate system that we can use to identify individual squares. For instance, square 3C is the first double-word square along the third row.

The players' names, racks, and scores are shown underneath the board. The rack of the side to move is given first, followed by the opponent's. Usually only the rack of the side to move is shown, since the opponent's rack is not known. In that case, the opponent's last move is given, because sometimes the mover can draw inferences from it. However, the position in the diagram is an endgame, so the opponent's tiles can be calculated exactly. In endgames we will show the opponent's tiles, in keeping with the principle of showing the reader all of the information that the mover might use to make his decision.

The tiles have face values. The "blank" tile, which can be used as any letter, is represented in the player's racks as a tile with a blank face with a point value of zero. In

the diagram, MAVEN's rack has a blank. A blank on the board is drawn as a letter with an outlined border and a point value of zero. In the diagram, there is a blank used as an E on square 10K.

Moves are shown by giving a word followed by a square, a score, and possibly information about the tiles left on the rack after the play. For example, MAVEN's move in the diagram was MOUTHPART (1A, 92+8), which indicates that the word MOUTHPART was played, starting at square 1A. The designation of the starting square as "1A" rather than "A1" indicates that the word was horizontal. The blank was played as a P, which accounts for the underlining of "P" in the notation, but we could have omitted the underline because MOUTHPART allows only one way to use the blank. In the rest of this thesis, the blank will be designated only when necessary, since the underlines are distracting. The "92" is the score of the play. Since MOUTHPART was the last move of the game, it also scored a bonus equal to double the sum of the face values of tiles left on the opponent's (Adam Logan's) rack, which accounts for the "+8" designation. MOUTHPART was a bingo, so there were no tiles remaining. Logan's previous play was QUAI (J2, 35, IIO), which means that the word QUAI was played, vertically down from J2, scoring 35 points, and leaving the tiles IIO on the rack.

Once a move is introduced into the text, we may abbreviate by omitting inessential information. For instance, we may remark that MOUTHPART was a crushing blow, leaving out the "(1A, 92+8)" part because we have already given it and there is only one possible place for MOUTHPART to play. Sometimes we may need to supply additional information to disambiguate moves, and the convention is to give only the starting square. For instance, we may comment on Adam Logan's previous play with language like, "QUAI (J2, 35, IIO) is definitely best. He should consider QUAI (M11, 26, IIIO), but QUAI (J2) is much better than QUAI (M11)."

2.2 Basic Aspects of the Space

This section presents statistics about the nature of high-level Scrabble. The goal is to build the reader's intuition, since later chapters will present models of the game based on this understanding.

2.2.1 Imperfect Information

Most of the Scrabble state space is imperfect information, because the opponent's tiles are unknown. However, the state space has perfect information in an important special case. The endgame is defined to be the phase of the game where the bag[3] contains no tiles, and therefore it is possible to calculate exactly which tiles remain on the opponent's rack.

The split of the state space into perfect and imperfect information components suggests that playing Scrabble requires search engines and evaluation functions that handle that distinction. Minimax search addresses perfect-information states, at least in principle.

[3] Definition: the bag is the set of tiles that have not yet entered the game. These are kept in a cloth bag, to shield their identities from the players. The bag is distinct from the concept of "unseen tiles," which would also include the tiles on the opponent's rack. The set of unseen tiles can be computed by subtracting the visible tiles (i.e., on the board or on your rack) from the initial pool of all tiles.

Programs can address imperfect-information states with combinations of shallow search and evaluation functions, but it is not obvious what to search, nor how to evaluate.

2.2.2 Non-Deterministic and Imperfect Information

The non-deterministic phase of the game has two extremes. At the start of the game, there are 100 tiles in the bag, and therefore the probabilities of what remains hidden shift almost continuously. At the end of the game, there are few unseen tiles, and therefore programs can completely enumerate aspects of the space.

The distinction between an almost continuous space and almost perfect-information space suggests that programs will use different methods on each. A simple linear evaluation model addresses broad imperfect-information states. Probability-weighted search addresses narrow imperfect-information states.

2.2.3 Gross Characteristics of Turns

In Scrabble the players alternate turns, so they have roughly an equal number of moves. This is said with the understanding that exchanges, passes, and lost challenges are turns that score zero points, rather than lost turns, as some people (must not be programmers!) do. The way MAVEN counts, the players have the same number of turns unless the player that moves first also moves last, in which case the first player has an extra turn.

Please note that scoring characteristics of turns depend on both the dictionary and the quality of the opposition. For example, players average higher scores in the United Kingdom than in North America. Are UK players better? The North Americans deny it, so the difference *must* be the dictionary; the UK's word list (SOWPODS) is a superset North America's TWL98. Table 2-1 shows the distribution of vocabulary by length within TWL98 and SOWPODS. In this thesis, the reference vocabulary is TWL98, unless specifically noted otherwise. Similarly, if the dictionary is held fixed then the level of scores rises with the quality of opposition. Since top humans are roughly comparable to MAVEN in strength, their scoring characteristics are similar. For example, MAVEN averages 34.3 points per turn, whereas top-flight humans average about 33 points.

Length	TWL98	SOWPODS
2	96	121
3	972	1229
4	3903	5155
5	8636	11812
6	15232	20964
7	23109	31229
8	28419	38043
9	24792	36845
10	20194	32178
11	15407	25225
12	11273	18366
13	7781	12563
14	5100	8115
15	3179	5002

Table 2-1 Vocabulary by Word Length

One technical point regarding point totals concerns how to apply end-of-game bonuses. In tournament games, when one player uses all of his tiles, he scores a bonus equal to double the sum of the tiles on his opponent's rack. But the rules printed in the game box state that at the end of the game the players are penalized by the sum of their unplayed tiles, and the side playing out scores a bonus equal to the sum of the opponent's tiles. For two-player games, these definitions result in identical point differentials, so it does not matter which definition you use. However, in multiplayer games there is a difference, so

MAVEN implements the rules as printed in the box. While point differentials are unchanged, point totals are higher under the tournament rules, so you must consider this difference if you compare MAVEN with other programs. Other programs may report that they average 35 points per turn, compared with MAVEN's 34.3, yet the results of the programs are identical.

Obviously, it is an advantage to move first, since having a 50-50 chance of an extra turn must have value. Perhaps that it is not truly obvious, since there could be a disadvantage to opening the board. To be completely accurate, the first turn does not score as well as the second turn on average, so there actually *is* compensation for moving second. Nevertheless, the first turn scores comfortably more than half of the score of an average turn, so moving first is an advantage. Moving first corresponds to a winning percentage of 56%, and a point differential of 14.2 points.

There is considerable variance in the score of a turn. The standard deviation is about 22 points per turn. When you consider that it is impossible to score less than zero points, it must strike you as surprising to have a standard deviation so high. What makes it possible is that bingos occur frequently. Thus, the distribution of scores is not symmetric about the mean. The distribution of scores in MAVEN-MAVEN games is roughly as follows:

1) Bingos, averaging 78.3 points, occur on 14.3% of turns.
2) Non-bingos average 27.0 points, with most between 18 and 40 points.
3) Nearly every game has a few non-bingo plays in the 40 to 55 point range, often using the J, Q, X, Z, or a combination of other heavy tiles, or playing to a triple-word square.
4) A few turns are dumps, which rid the rack of awkward tiles while scoring from 8 to 16 points.
5) Exchanges happen with awful racks, about 1.7% of all turns.

2.2.4 Gross Characteristics of Games

Computer games average 11.7 turns per player, or 23.4 for a whole game. Human games are somewhat longer, averaging about 12.5 turns per player. Both humans and computers average about 400 points per player per game.

The standard deviation in the difference in scores of a whole game is about 95 points. (Note that this is not the same as the standard deviation of the score of a single player.) Accordingly, most Scrabble games are not close. It is common for one side to score a couple of quick bingos and then coast to a win while the other side struggles.

For the same reason, only the largest leads are safe. Even when a game is half over, the standard deviation remains about 65 points, so it is possible to come back. One bingo will make you a favorite, and it is even possible to come back from such a deficit without a bingo. The leader could be forced to exchange a bad rack, or could be stuck with the Q, or the trailer could score well using J, Q, X, or Z.

2.2.5 Scoring as a Function of Stage of Game

Table 2-2 shows how scoring changes over the course of the game. The first turn averages 28.8 points, and the second turn (i.e., the second player's first move) averages 34.7. The high average of the second turn is due to the many opportunities to play bingos on a wide-open board after the first turn. Scores are high for a while and then drift downward as the board clogs up.

You can see that bingo frequency accounts for most of the difference in scores. The average scores of both bingos and non-bingos is a remarkably constant function of the turn number. The average score of non-bingos remains relatively constant until a sudden decline around turn 17. At that point, many games go into the endgame phase.

Most games are finished by move 25, so all statistics after that reflect the subset of games that happen to last that long. For instance, the board may have been very constricted. Or one player may have been stuck with the Q, which leads to long games because it is in the interest of the other player to play out one tile at a time. The extremely low frequency of bingos past move 25 reflects the fact that such positions are almost always endgames.

Turn	Mean	Bingo%	Bingo	Other
1	28.8	12.7%	75.3	21.8
2	34.7	20.3%	75.7	24.2
3	39.4	23.2%	77.0	27.9
4	39.8	20.4%	77.9	29.6
5	40.0	19.9%	78.5	30.3
6	39.9	19.1%	78.8	30.8
7	40.2	18.4%	78.9	31.0
8	39.8	17.9%	79.1	31.0
9	39.7	17.5%	79.2	31.1
10	39.2	17.1%	79.1	31.1
11	39.0	16.5%	78.9	31.0
12	38.5	15.9%	79.1	30.8
13	38.0	15.7%	79.2	30.7
14	37.9	15.3%	79.1	30.3
15	37.2	14.7%	78.8	30.1
16	36.4	13.7%	78.9	29.8
17	35.4	12.9%	78.2	29.1
18	33.7	11.8%	78.5	27.9
19	31.2	10.0%	77.4	26.6
20	28.6	8.6%	77.2	24.7
21	25.3	6.4%	76.9	22.4
22	22.0	4.4%	76.4	19.8
23	19.3	2.7%	76.2	17.2
24	16.6	1.5%	74.5	15.2
25	14.2	1.0%	77.0	13.7
26	11.9	0.7%	73.9	11.7
27	10.1	0.2%	71.8	10.3
28	8.5	0.1%	74.3	8.7
29	7.7	0.09%	101	7.4
30	6.9	0.07%	74	5.8

Table 2-2 Score as a Function of Turn

2.2.6 Scoring as a Function of Tiles

An enlightening view of scoring is to see how it depends on the tiles you possess in the rack. Table 2-3 shows the average score of moves[4] as a function of tiles. The "Tile in Rack" column shows the average score given that a specific tile is in the rack. Notice that the frequent tiles (AEIONRT) have scores that are around average, but the infrequent tiles (especially Blank and Q) strongly affect the score. This is to be expected; frequent tiles are present on most racks, so their average must be close to the average play, whereas infrequent tiles can affect the score of a small number of racks.

[4] Like all data in this chapter, these are from MAVEN-MAVEN games.

Another thing you can gather from the "Tile in Rack" column is a rough ordering of the quality of tiles. Obviously, the blank, S, Z, and X are standouts, whereas Q, V, U, W, and G are millstones.

The "Two in Rack" column shows how holding duplicates of tiles affects scoring. Of course, the duplicate blank is excellent. We see that duplicate S is just like a single S, duplicate E is bearable, and every remaining duplicate tile reduces scoring.

The "When Played" column shows how the score of a play depends on the tiles used to make that play. Again, the good tiles are evident. Moves that use Blank, S, X, and Z have above-average scores,[5] whereas moves using Q, V, W, U, and G have poor average scores. This confirms previous assertions.

Another group of tiles is interesting because of a difference between the "Tile in Rack" and "When Played" columns. The tiles A, E, N, R, and T have high values when played, whereas they were near average when in the rack. What can we make of that? The explanation is that these are bingo-making tiles. They tend to be saved for bingos. This raises the average score when they *play*, but their average score when *held* is normal because of the turns spent sitting in the rack.

The Tile	Tile in Rack	Two in Rack	When Played
?	48.5	67.9	70.6
A	36.4	31.7	39.4
B	34.2	30.2	35.5
C	35.9	30.4	40.9
D	36.1	31.6	39.5
E	36.6	34.5	41.8
F	33.9	33.6	35.3
G	33.2	27.6	35.2
H	37.2	32.6	39.4
I	35.3	28.9	38.5
J	34.0	-	35.6
K	35.1	-	36.6
L	35.2	29.6	39.1
M	36.4	32.7	39.3
N	36.0	30.9	40.9
O	34.7	29.5	36.6
P	35.6	31.4	38.4
Q	27.4	-	30.7
R	36.5	31.4	42.6
S	39.3	39.4	51.3
T	36.0	31.7	40.3
U	32.9	24.7	35.4
V	31.7	24.1	32.7
W	33.1	28.0	33.8
X	38.6	-	43.0
Y	35.2	29.7	36.1
Z	38.4	-	43.7

Table 2-3 Score as a Function of Rack

These views show how important it is to have good tiles. For instance, moves that use a Blank average 70.6 points, which means that almost all are bingos. If one side draws both Blanks, then what chance has the other side of winning? Tournament players win only 1 game out of 4 when the opponent draws both Blanks. Fortunately, the Blanks split evenly in almost 50% of the games.

[5] One technical detail to note is a bias regarding "average." The scores in the "When Played" column are higher than you might expect because the scores in this table count for each tile played, which increases the weight of long words relative to short words. This is OK, as long as we do not compare a 40 point average in this table to a 40 point average elsewhere.

2.2.7 Scoring as a Function of Vocabulary

The tables in this section show how scoring is distributed as a function of the words used in the game. Some words are more important than others, either because they play more frequently, or for higher scores, or both.

Table 2-4 shows how scoring is distributed as a function of the tile turnover (i.e., tiles played). The top row, showing data for 7 tiles of turnover, tell us about bingos. While bingos are 14.3% of all moves, they account for 20% of all scoring. This confirms the importance of bingos.

Tiles	Freq	Score
7	14.3%	78.3
6	6.7%	32.9
5	18.3%	30.5
4	24.7%	28.5
3	21.5%	26.8
2	10.3%	23.2
1	2.6%	11.5

Table 2-4 Score by Tile Turnover

The table shows that the average move turns over 4.2 tiles. This confirms that games should end in a little less than 100 / 4.2 = 23.8 turns, where the "little less" is required because at least one player must be stuck with at least one tile at the end of a game.

The observation that score increases with tile turnover influenced pre-computer strategists to try to play more tiles per turn. This is a little like believing that playing basketball will make you taller, and is further explored in Chapter 5.

Length	Freq	Score	Per Game
2	5.2%	22.1	13.2
3	16.9%	24.2	47.0
4	22.9%	27.2	71.6
5	23.0%	29.4	77.8
6	12.9%	29.7	44.1
7	9.4%	59.4	64.2
8	9.2%	75.4	79.8
>= 9	0.5%	64.5	3.7

Table 2-5 Score by Word Length

Table 2-5 shows how plays vary by length of word, along with scoring characteristics. Note that words up to 5 letters constitute 68% of all words played, despite TWL98 having only 13,607 such words! This illustrates an important attribute of the game: it is relatively easy to gain skill up to a point. If a player masters the short words alone then he will play most moves correctly. Alas, the short words do not account for most of the difference in skill between players. Genuine mastery only accrues to those who multiply their efforts six-fold, to cover the full 80,367 words of length up to 8 letters in TWL98. In addition to the larger number of words, it must be considered that anagramming the eight-letter words is harder than anagramming the fives.

JQXZ words are important. In fact, the 18 most useful words are all JQXZ words.[6] An average game contains 4.5 JQXZ words. Table 2-6 shows how the top ten score. Note that QAT accounts for almost 1% of all the words played in a game, and these ten words account for 2.4% of the words.

The column labeled "Score" in Table 2-6 shows the average score achieved by each word. Pairwise comparisons of the average scores reveals insights into rack evaluation. For example, note that moves that play X score more than moves that play J, Q, and Z. This reflects

Word	Freq	Score
QAT	0.73%	24.2
QAID	0.29%	27.2
XI	0.25%	38.3
EX	0.20%	40.3
OX	0.20%	40.7
JO	0.17%	29.1
AX	0.15%	40.9
SUQ	0.15%	29.9
QUA	0.14%	26.9
XU	0.14%	36.2

Table 2-6 Top Ten JQXZ

[6] The only words in the top 44 that don't contain JQXZ are OF, IF, EF, and FA.

the value of an X. The table shows 5 plays that use X: AX, EX, XI, OX, and XU. Since each vowel is represented, we can learn something about the value of each vowel. For instance, moves XU plays average only 36.2 points, whereas AX averages 40.9. We might infer that A is more valuable than U, and we would be right. Another interesting comparison is QAT versus QATS. The presence of an S increases the average score from 24.2 to 33.0. Obviously, an S is a terrific tile.

Yet such comparisons do not tell the whole truth. For example the most valuable vowel is actually E, whereas the table shows EX averaging 40.3 points against AX's 40.9 points. EX averages less because EX implies that the player has no choice. For example, it could be that the E is the only vowel on the rack, or because the player is clearing a duplicate E. This tends to lower the average scores for EX. To see the whole truth, you must do more complicated analysis.

The foregoing tables show how categories of vocabulary affect scoring. Another dimension is *utility*. Some words occur more often than others do. Some words (bingos, in particular) score more than others do. Table 2-7 shows how scoring is distributed within the SOWPODS vocabulary[7]. In a run of 131,000 games, there were 113,798 distinct words played. The most useful 11,380 words accounted for 53.1% of the scoring.

Utility Decile	Word Count	Freq	Scoring Percentage
Most useful	11380	62.5%	53.1%
90%	22760	75.6%	68.3%
80%	34139	83.5%	78.2%
70%	45519	88.9%	85.3%
60%	56899	92.8%	90.5%
50%	68279	95.7%	94.2%
40%	79659	97.6%	96.8%
30%	91038	98.9%	98.5%
20%	102418	99.5%	99.5%
Least useful	113798	100%	100%

Table 2-7 Score by Word Frequency

This statistic confirms that the threshold of expertise is tantalizingly low, because mastering 11,380 words is an achievable goal for any aspiring player. The rest of the table shows how difficult genuine mastery is. Suppose that a player has true mastery (to the extent that he never misses a move) of the most useful 56,899 words, but is unfamiliar with the rest of the words. Then he would only find 92.8% of the best plays. It is unclear what this would cost in score, because the table does not capture the quality of second-best moves.

The most useful 10% of words are mostly short words and high-frequency bingos. This should surprise no one.

You might be surprised at the identity of the least useful words. One's first instinct is that the least useful words are low-frequency bingos. Actually, they tend to be non-bingo plurals of short words that contain high-frequency tiles. This makes sense, when you think about it. Such words are rarely best plays, and when they are there is usually a second-best play that is almost as good. So one would lose little by not knowing such a word.

[7] TWL98 is expected to show a similar distribution.

2.2.8 Scoring as a Function of Position

Finally, here are two positional views of the game. Figure 2-8 shows how often tiles cover each square at the end of the game. MAVEN's first move is always horizontal through the center square, which accounts for the square being solid black, and also for the relatively dark squares along the center row. You can see how games tend to evolve along the major diagonals, where the double-word squares reside. You can also see how play "stretches" to cover the premium squares. If your vision is very sharp, you may detect that play tends to evolve more down and to the right.

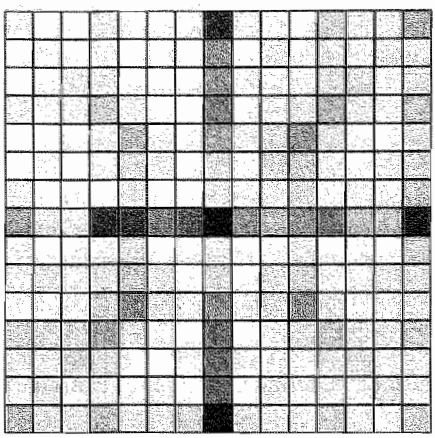


Figure 2-8 Usage of Square.

Darker is More Frequent.

Figure 2-9 shows the average score of moves covering each square. The diagram makes clear how the use of premium squares dominates scoring. You can see the dark squares near the triple word squares at the center of each side. The corner triple word squares also have dark squares, though not as dark as the center triples. You can also make out a box near the double-double region. Most of the rest of the board is fairly light, including the double-word squares. As a general rule, double word squares are not, by themselves, significant contributors to big scores.

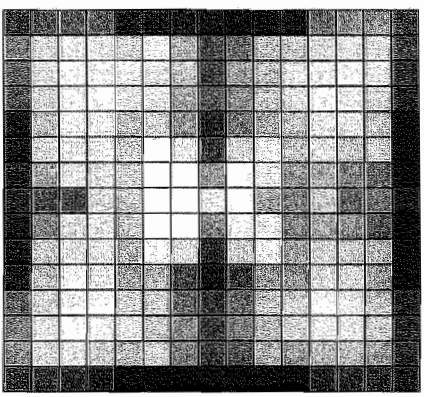


Figure 2-9 Score by Square.

Darker is Higher Scoring.

2.2.9 Summary

The preceding subsections give you a good overview of the game. You have seen:

1) How the score of a turn varies, in general.
2) How the variance of turns affects the competitive balance of the game.
3) How the score of moves depends on tiles.
4) How the words affect scoring.
5) How the premium squares influence turns.

Besides contributing to your understanding of Scrabble as an abstract space, the author hopes that you are impressed by the exceptional balance of the game. Scrabble is a masterpiece of game design, in which balanced, conflicting forces interact to create and maintain competitive tension.

2.3 Human Scrabble Strategy

The human cognitive process that finds high-scoring plays differs from the computer algorithms that generate all plays. For a human the process involves anagramming the rack and looking for hotspots.[8] Many techniques help experts to carry out this process.

2.3.1 Vocabulary

Adult native speakers of English typically have vocabularies of about 25,000 words. Alas, that vocabulary will not get anywhere over a Scrabble board. The obvious shortcoming is that the vocabulary is too small. Top class Scrabble masters need to recognize all words up to eight letters, and there are 80,367 such words. Another deficiency is that normal vocabulary does not include the weird, short words that are vitally important to skillful play, as described by the previous section. Therefore, the first task of fledgling experts is to learn the words.

Humans generally learn high-frequency words first. The National Scrabble Association (NSA) publishes the Cheat Sheet, a single sheet of paper that lists all of the 2-letter and 3-letter words, plus the vowel dumps and short JQXZ words. Simply mastering the words on the Cheat Sheet will turn a family player into a tournament player. However, of course there are many more words to learn.

Generally, human experts that are inclined to work systematically will learn words in categories. For example, they may learn the 7-letter words having the letters RETINA. Lists of words from important categories are available from several sources.

However, learning words within categories has limitations. First, people have not created lists that cover the low-frequency words, so this approach will only take you so far. In addition, even some high-frequency words manage to slip into the cracks between published categories.

Ultimately, if you want to learn the words then you must read the dictionary (or another exhaustive source). This has the advantage that you learn definitions, which improves retention. For high efficiency, most players use a highlighter pen to mark the words that are uncommon, so they do not spend time refreshing familiar words.

Another key concept, introduced by Charlie Carroll and published by Nick Ballard [5], is to learn the words in *frequency* order. In this strategy, a player commits to learning, say, all of the seven letter words. He accomplishes this by scanning the words in order of frequency of getting that word in a seven-tile draw from the bag. For example, the relative frequency of AEEINRT is 9 * 12 * 11/2 * 9 * 6 * 6 * 6, and the word FREESIA has frequency 2 * 6 * 12 * 11/2 * 4 * 9 * 9, which is only 2/9 as often as TRAINEE. (See Appendix A for a table of the tile frequencies.)

There are two advantages to learning words in frequency order. First is that you learn the most *useful* words first, rather than by category. Second is that you have a method of confidently challenging unfamiliar words. For example, suppose that an opponent plays MARINET against you, and you have memorized the words down to FREESIA.

[8] A hotspot is a place on the board where high-scoring moves are likely. Hotspots frequently involve premium squares.

MARINET is unfamiliar, so you can challenge with 100% confidence. The reason is that MARINET has frequency 2 * 9 * 6 * 9 * 6 * 12 * 8, which is greater than FREESIA, so if MARINET is unfamiliar then it must be phony.[9]

2.3.2 Anagramming

Recognizing legal words is only the beginning. You still need to find the plays, especially the bingos. The process of rearranging letters to make words is called anagramming, and it is the fundamental skill of expert players.

General anagramming skill consists of the ability to mentally rearrange racks in all possible orders in one's mind. However, no human can really go through all 5040 permutations of seven tiles, so every human makes some approximation to the ideal behavior. Generally, the approximation has something to do with prefixes and suffixes.

Humans look for 3-letter prefixes or suffixes that they can make out of their tiles. They physically move these to one end of their rack, and mentally anagram the remaining tiles. Let us take the example of DFGIINN. A human would see the –ING suffix, and move these tiles to the end of the rack, to form DFIN-ING. Then he would mentally shuffle the remaining tiles, and the word FINDING is easy to see. An expert can carry out this process in about a second.

Anagramming using prefixes and suffixes is more efficient than you might expect. The top 267 prefixes and suffixes (see Table 2-10) are contained in 80% of all seven and eight letter words. Therefore, if you can carry out the technique perfectly then you expect to find at least 80% of the legal bingos.

However, brute-force anagramming is too slow for many purposes. Some racks scream for a Q play, for instance. In that situation, it is helpful to be able to run down a list of high-frequency Q words. Most humans have memorized such lists.

Additionally, humans have mnemonic tricks for recalling the anagrams of certain words. *One cute idea is to "fix-up" a case where prefix search does not work.* For example, consider the rack ADNOSTU. Prefix search turns up OUT + SAND, but OUTSAND is phony. The trick is to memorize that OUT + SAND = ASTOUND. Thus, a failure in prefix search is converted to a success.

[9] Should have played MINARET or RAIMENT!

Tile	Suffixes	Prefixes
B	BLE	SUB BRA BAR BRO BRI BUR BED BLA BLO BLU BAN BAC BOO BEA
C	NIC RIC TIC ICS CES NCE CAL CTS	CON CAR CRA COR CRO CAL CLA REC CRE COL CLO CAN CAT INC CRI COA
D	ED DES OID IDS ARD NDS RDS IDE AND EAD	DIS DEC RED DIA IND DRA DES
F	FUL IFY	FOR FLA REF FE DEF FLO INF
G	ING NGS GES AGE	GRA GRI GRO REG GAL
H	ISH HES	CHA CHI CHE CHO SHA SHE SHI SHO HAL HAN HAR HEA
K	IKE CKS OCK ACK	SKI
L	LES ALS ELS IAL ILS LAS NAL LLS OLE OLS TAL LET ILE	SAL SOL REL LEA LIN LAN
M	MEN MAN ISM UMS IUM MES SMS MIC ORM	MIS MAN MON MIN COM MAR MOR IMP MIL REM
N	ONS INS NES INE ION ENS ANS ONE NASIAN ANT	NON INS SAN SNO
P	UPS	PRE PRO PAR PLA PER REP PRI POL PIN PAN PEN EPI PAL PLU CAP
Q		QUI QUA SQU
R	ER ERS RES ORS ARS TOR RTS RAS LAR RAL	RES TRI STR TRA RET REA TRO SER TRE AIR TER
S	IES ISE ESS SES IAS EES OSE OES SIS	SEA ISO CAS SCA SCO SCR SPA SPE SPI SPO POS
T	EST ATE TES ETS IST STS ITE ENT NTS ITS OTS ATS	STA STE STI STO ANT ENT INT TEN
U	OUS URE UTS URS UES	OUT UNS UNC CUR CRU COU TUR TRU SUN
V	IVE VES	OVE REV VER
W	WAY OWS	WHI WIN WAR SWA SWI WO
X	XES	
Y	ILY ITY ERY LLY AYS DLY ELY EYS ARY BLY TLY	
Z	IZE ZES	

Table 2-10 Prefixes and Suffixes

A popular trick is to memorize the letters that go along with a "stem" word to make a bingo. For example, the six-letter stem RETINA makes a bingo with CDEFGHIKLMNPRSTUW. If you remember this, then if you have RETINA + B you can avoid looking for bingos entirely, and if an opponent tries to slip BANTIER past you then you can challenge with 100% confidence.

While you can memorize the letters that go along with each stem by rote, there is a more efficient way. You can remember the letters that bingo with RETINA by recalling the phrase THE RED PUPIL PREFERS MUCH WINKING. This phrase is called an *anamonic*, since it is a mnemonic for

Stem	Anamonic
TORIES	BUNCH OF OLD GRUMPS
LADIES	LOVELY BRIDES MUST NOT HOP
INMATE	RELAXING BY HIS CELL DOOR
ATTIRE	CAP, SCARF, AND BELT

Table 2-11 Anamonics

anagrams. It is easy for a native English speaker to remember this phrase, because of the association between RETINA and PUPIL, which are both parts of the eye. Table 2-11 shows a few more anamonics.

Human experts have devised anamonics for at least the 200 most frequent six-letter stems, and the 500 most frequent seven-letter stems, covering over 10,000 letter combinations including those of highest frequency [72]. Players who master anamonics seldom miss high-frequency bingos. What is more they play quickly because they do not spend time looking for bingos that are not there.

An important technique is to make up a list of all words not covered by any tricks. These are the difficult anagrams, which include such beauties as AEEIINRT, which is one of the most important bingos of all (INERTIAE, which is known to have come up twice in a single game). These must be practiced relentlessly.

Of course, humans do practice constantly. Humans make up flashcards listing ten anagram questions on one side of the card, and the answers on the other. With practice, they can go through the whole card in less than 30 seconds. If they happen to get a question wrong, then they stick that card into the middle of the deck. Otherwise, they stick that card at the end of the deck.

Because 2/3 of all bingos in human games use a blank, it is important to practice anagramming with blanks. Of particular interest are racks in which the blank can be used for only a single letter. For example, the rack ?AEIORT is a so-called *unistem*, because the only word that matches is EROTICA. Unistems are particularly important to practice, because there is likely to be only one legal bingo, so you stand to lose a lot by missing it.

Some humans use computer programs for practice. Flashcard programs [6] can emulate a deck of cards, and provide extra functionality, like showing related words and other mnemonic tricks. It can also track a user's training record, and bias training towards words that are likely to be missed.

The principles that work for finding bingos also apply to other categories of words.

You are probably impressed by the ingenuity that humans apply to this difficult task. Indeed, humans perform better than you might expect. However, keep in mind that humans need these tricks because unaided human cognitive capabilities are hopelessly inadequate for the task. Even with these tricks, humans still miss plenty of moves.

2.3.3 Hotspots

A *hotspot* is a place on the board where productive scores are possible. Of course, some hotspots require specific tiles to create the high score, in which case we speak of a "hotspot for an F," for instance.

Hotspots are important to human move generation, since it helps focus the search in two ways. First, humans save time by only considering spots where productive scores are possible. Second, the hotspot often provides clues as to where specific tiles should go. For instance, the Q should go on the Triple Letter Square, or only an M can play adjacent to the Y from the board.

A *bingo line* is a spot where a bingo can land. A bingo line needs seven empty squares and a minimum of contact with the board. The quality of a bingo line depends on how many words play there.

A *triple-word line* is a hotspot that allows a move to cover a triple word square. Tripling the value of the tiles played will usually yield an above-average score.

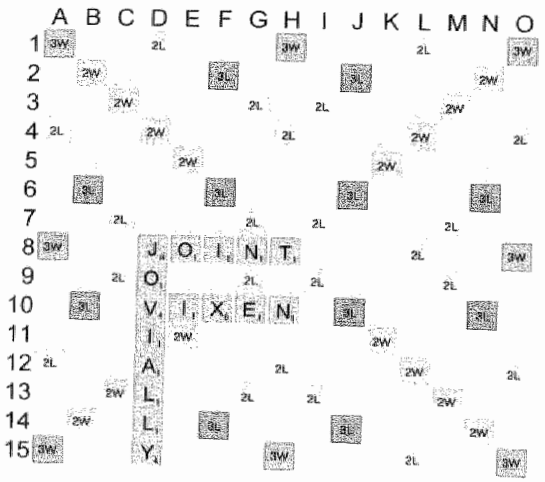
A *hook* is a single-tile extension to a word on the board, which is used in conjunction with a move going in the other direction. The most common hook tile is an S, used to pluralize a word, and any tile (even Q and Z) can be a hook tile if the right word is on the board.

An *overlap* is a move that scores points using crosswords.

Some hotspots are hot simply by virtue of the quantity of premium squares located there. For example, we speak of a double-double spot for a hotspot that contains two double word squares. Similarly, there are DLS-DWS spots, where a double letter square and a double word square reside.

An *extension* is a move that adds several tiles to a word already on the board, usually to hit a word multiplier.

Table 2-12 and Position 2-3 illustrate types of hotspots.



Position 2-2 Hotspots Illustrated

Type	Examples
8 LB	Through JOVIALLY.
	15A triple-triple.
7 LB	7LB overlap JOINT
	or VIXEN on row 7,
	9, or 11, or column I.
TWS	15A or H10
Hook	JOINTS or VIXENS
Overlap	Row 11, using 11E.
	Column C, using C13.
Bonus	E5 double-double
	14B DWS-TLS
	13C DWS-DLS
Extension	8A CONJOINT
	8A DISJOINT

Table 2-12 Hotspots

2.3.4 Evaluation

Of course, a human will find more than one move. How does he decide which is better?

Obviously, the score is the most important factor. Usually the question will be whether thedifference in score is compensated by qualitative advantages of the lower-scoring move.

2.3.4.1 Heuristic Judgment

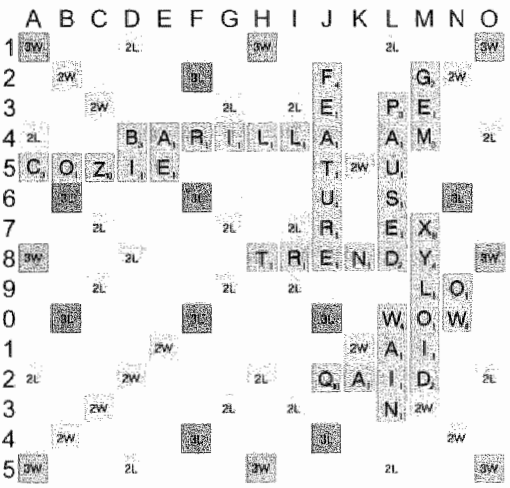
Normally the judgment is purely heuristic. The human isolates the difference between the plays, and only evaluates the tradeoff. For example, one word might play an E, whereas the other plays an A. Such a difference might be worth, say, 4 points to the move that plays the A (and keeps the E).

In the case of rack evaluations, a human will probably make conscious tradeoffs of points versus tiles. However, sometimes the judgment is hard to render into units of points. For example, consider a player trying to protect his lead. He might have a high scoring play and a low scoring play blocking a hotspot that the opponent might use to come from behind. Which is better? It is hard to give rules of thumb.

2.3.4.2 Search

Search resolves some situations. The obvious case is when the bag is empty, since we can deduce the opponent's rack and compute the best tactics. When the bag is not empty, it may be possible to foresee specific outcomes. An offensive instance is called a *setup*. In a setup, you foresee that the tiles you keep play productively after your move. A defensive instance is called a *block*. You foresee that a spot on the board benefits the opponent, and play to interfere with it.

Humans can search spaces that computers cannot easily search. For example, humans might evaluate a tradeoff by breaking the set of positions into two cases: those in which the opponent holds an S, and those where he does not. Then the player evaluates the risks and rewards in both cases. This is a search of a *conceptual space*. Consider the Position 2-3, which first appeared in *Medleys* [7]. Which move has a higher point differential: OBSCURED (A2, 86), or the volatile ROSEBUD (O8,98), which hangs a double-crossed triple word square?



Position 2-13 Conceptual Search

One expert's conceptual search of this position consisted of the following comment to *Medleys*: ROSEBUD is "horrible. Very few extra points enables a homicidal reply. What if Opp has ZITS, QATS, JOTS, or bingo?" Ah, the frequently useful zits, qats, jots and bingo search space. This example shows that the mere fact that humans are capable of conceptual search does not imply that they can actually carry it out. Someone should tell this player that the Q and Z are already on the

board, and if the opponent held a J he would have played JOWL(H1, 42) instead of OW (N9, 32).

This example of bad human reasoning emphasizes that there is a difference between conceptual search and paranoia. Every position allows devastation if the opponent holds perfect tiles. For example, a search of the "AGJINNU" space results in a strong preference for OBSCURED, since it blocks opponent's JAUNCING (A1, 221).

Randy Hersom's conceptual search resulted in the following comment: "The difference between leaving a double-crossed S and a direct triple, possibly a triple-triple, is not 12 points." Hersom is breaking the position down into a choice between leaving the A-column open (where a triple-triple could possibly land) and leaving the big hook after ROSEBUD. He concludes that ROSEBUD will have a higher differential.

The evaluation of ROSEBUD depends on which side gets the first S, and how many points they will score by using the TWS at 15-O. It is actually possible to calculate a reasonable value for the cost of ROSEBUD's big hotspot. The basic formula is

Value = Extra Points Scored * (Chance you get it – Chance opponent gets it)

To calculate the components of this relation is fairly involved, and perhaps beyond the scope of this thesis. Table 2-14 shows, without justification, the steps in the calculation.

Name	Meaning	Formulas	Value
K	Number of Key tiles (S, here)		3
U	Number of Unseen		49
S	Score in spot	5 tile play, ending in S at 15-O	72
M	Average score in position		40
R	Rack value of key tile		8
X	Extra points scored in spot	S – M – R	24
A	Probability of drawing S	K / (U – 3)	0.0652
P	Probability opponent has S now.	7 * A * (1 – 3 * A + 5 * A * A)	0.3769
E	Opponent's chance of getting S first.	P / (2 – P)	0.2322
V	Value of opening	X * E	5.57 points

Table 2-14 Example of a "Key Tile" Conceptual Search

In English: ROSEBUD's point differential loses 5.57 points because of the hook it creates. Since ROSEBUD outscores OBSCURED by 12 points, ROSEBUD has higher average point differential. Simulations confirm that ROSEBUD has about a 7-point advantage over OBSCURED.

While perhaps no human could carry out the calculation above during a game, some players have an uncanny ability to evaluate hotspots because they have solved many similar problems in the past. Charlie Carroll is one such player. He appraised ROSEBUD accurately and wrote, "Most people focus on the immediate effects of a huge opening,

forgetting that they might be the one to cash the opening. Another point people miss is that the openness of the rest of the board has a large effect on the wisdom of this type of play."

2.3.4.3 Trickery

Sometimes humans can resort to desperation plays akin to throwing a Hail Mary touchdown pass. Usually a search has indicated that against good play by the opponent there is no way to win, but one alternative allows a comeback if the opponent misplays.

For example, if you trail by less than a bingo going into the endgame, you can try to *fish* a bingo out of the bag. For example, you might play one tile, keeping RETINA. Against a savvy opponent who knows what you are up to, this might be a futile gesture. On the bright side, the opponent may be unable to respond to all possible places where you might play a bingo.

2.3.5 Human Expertise: Putting it all Together

Human experts always have a focused move generation strategy, usually consisting of mentally scanning a list for words that play key tiles. They supplement that with shuffling the tiles for prefix and suffix searching.

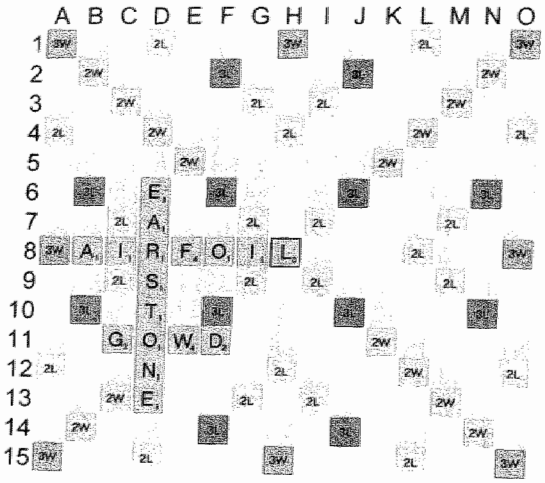
Human move generation focuses the search effort by seeking moves that exploit hotspots. Once they find a good move, they can focus their search by trying to improve upon that move. For instance, given a 30-point move, they might restrict their attention to spots that could potentially yield more than that.

While generating plays, they weigh the pros and cons of moves, with a heavy emphasis on scoring points, and secondary emphasis on keeping good tiles. If two moves affect the board differently then they may try to factor that impact into the comparison as well. Usually the judgment is heuristic, but search may be involved in specific cases.

After a human searches for a while he may switch attention to asking whether a different move could surpass the one he has found. In this mode of thought, the player tries to prove that no other spot could yield more than he has found already.

2.3.6 Example

We will illustrate the process using Position 2-3. In this position, the player is Adam Logan, North American Champion in 1996. He holds several awkward tiles. Nevertheless, experience teaches that it is always worth checking for a bingo. Using the prefix and suffix table, we try the possibilities shown in Table 2-15. There are (probably) no bingos.



Logan: E, E, G, L, U, U, V, 66

MAVEN's last: GOWD (11C, 18) 88

Position 2-3 Human Move Selection

Next, we consider dumping a few bad tiles. We have a duplicate U (very bad) and a V (bad). Our duplicate E and G are poor. The L is so-so. We would like to keep an E, which is a good tile. See any plays that dump the EUV? Yes, UVEA from B5 is an automatic find for human experts because UVEA is a "3-vowelled-4," which all experts would know by heart. (They might not know what it means, though!)

Particle	Spot	Anagram	Find?
-AGE	B3	ELUUV-AGE	No
-ILE	G3	EGUUV-ILE	No
-IVE	G3	EGLUU-IVE	No
Try –ULE,	Any	?EGUV-ULE	No

Table 2-15 Prefix and Suffix Search

UVEA scores 15, and keeps EGLU. This is not great, so we look for more. Having studied the "4-to-make-5" list, we know that UVEA takes a back L hook, so UVEAL from B5 is possible, scoring 16 and keeping EGU. UVEAL seems slightly superior to UVEA, because it scores an extra point, and playing the L should garner a 1.5-point improvement in rack. Why 1.5 points? You will find out in Chapter 5!

Against UVEAL is the two-vowel, two-consonant rack of EGLU versus the two-vowel, one-consonant rack of EGU, where the 2-2 balance is obviously better. Vowel-consonant balance of 2-2 is valued at 0.5, and a 2-1 balance is –0.5, so UVEAL takes a 1-point hit. Figure UVEAL to be 16 – 15 (for score) + 1.5 (for eliminating the L) – 1 (for worse vowel/consonant balance) = 1.5 points better than UVEA.

The "3-vowelled-5" list produces UVULA, which plays from B4. We should pluralize it to make UVULAE, which scores an extra point and eliminates the duplicated E. UVULAE scores 11, keeping EG. Normally I would rate UVULAE as slightly better than UVEAL, because ridding the rack of the other U is worth 4.5 points, and retaining a 1-1 count of vowels and consonants is worth 1 point over UVEAL's EGU. That makes a 5.5 point edge, whereas UVULAE scores 4 points less. However, UVULAE has a positional downside: it opens the A1 triple-word square by allowing a big overlap on the A4 double-letter square. An opponent need only hold an N to get a decent score (e.g., FLAN (A1) scores 27), and an M would yield a great score (FLAM (A1) scores 43), and do not

even contemplate what happens when the opponent holds X. (OK, put away that calculator. FLAX scores 83. XU is a currency unit of Vietnam.) The opening to A1 is not the only defensive liability. The opponent could now be sitting with an unplayable Q, and jumping for joy because he can dump his Q from 4A, scoring over 46 points because he hits the double word square at 4D. With UVULAE projected to hold only a 1.5-point edge over UVEAL, it seems that the defensive liabilities are too large. The triple-word square exposure is over 3 points by itself. (Simulation (chapter 10) shows that the actual defensive liability is about 5 points.)

The question remains whether anything can beat UVEAL. In this position, probably nothing does. The V does not make any two-letter words, so hotspots that involve overlaps are not feasible. The G only overlaps an A (for AG) or O (for GO), and these opportunities do not exist on the board (e.g., GLEE 7F is only 14 points, and keeps the hideous tiles UUV). UVEAL gets 3 times the V, and it seems impossible to get 4 times the V. Because the penalty from keeping a duplicate U is so heavy (about 12 points), we are forced to play at least one U this turn. It is hard to imagine any better rack leave than EGU, EGLU, or EG. There is no way to play these same tiles at another spot on the board. So UVEAL is the best play.

In the game, Logan chose UVEA, probably judging that vowel/consonant balance is more significant than stated above. Simulations give UVEAL an edge of about a tenth of a point over UVEA, so we can regard these two as equal. UVULAE is 3.7 points back. The human process works well here.

2.4 Comparison of Computer and Human Expertise

Human masters are excellent players—far better than you might expect after considering human cognitive limitations. Appendix B gives annotated games that show how accurate they can be.

Despite the ingenuity of human masters, human Scrabble strategy should have a tough time against expert computer programs. The difficulty is that human move generation skills are deficient. If a human misses bingos in even 20% of his games, that will add up against computers that never miss. Add in the inevitable errors on non-bingos, and the best humans would face a deficit of 15 to 20 points per game.

It is questionable whether humans have any tricks in their arsenal to counterbalance that disadvantage. The end-of-game fishing trick has potential to swing about 5% of games, and humans can count on endgame errors for additional compensation. Nevertheless, it is doubtful that total amounts to 20 points per game.

2.4.1 Validation against Wapnick's Book

When MAVEN's drive to achieve world dominance started, the author obtained a copy of the book *The Champion's Guide to Winning at Scrabble*, by Joel Wapnick. This book, sadly out of print,[10] is to this day the deepest exposition of the game. Joel Wapnick won the North American Championship in 1983 and was runner-up in 1992, and won the

[10] Actually, there is an electronic second edition [26], in which Wapnick reexamines the theories of the first edition in light of computer data. MAVEN simulations play a large role in the revisions, and this author is grateful that Joel was able to benefit from this research.

World Championship in 1999 and was runner-up in 1993 and 2001, so he is amply qualified to enlighten aspiring experts.

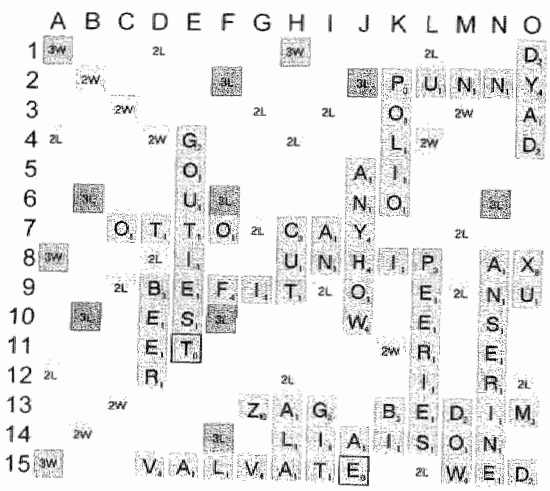
MAVEN went through all of the examples in the book. MAVEN compared the move recommended by Wapnick, or the move made in a game by an actual player, with its own move. There were over 50 sample positions and 8 fully annotated games, covering all phases of the game, with an emphasis on the fine points. The author concluded that

1) computer play should be superior through the early game, contingent solely upon the program's ability to evaluate rack leaves, and

2) human weakness in move generation continues through the pre-endgame and endgame, but there are significant difficulties in evaluation for this phase, so naïve computer programs are at best break-even against top masters.

For a typical example of human weakness, consider the opening rack of EHJOPSS. One player played JOSHES (8D), which scored 48 points. Unfortunately, he missed JOSEPHS, a 104-point bingo. JOSEPHS rarely occurs, but when it does, it is well worth finding. This example is typical; the word is infrequent and the player probably never studies that far down in the frequency list. Humans do not even notice such errors unless a computer checks the moves.

For a typical example of human strength, consider Position 2-4, which occurred first in *Letters for Expert Game Players*, a newsletter published in the 1980's. Wapnick reprinted it in his book, and it is so remarkable that the author feels compelled to publish it once more [8]. Please note that this position occurred before 1993, when QAT was introduced to the lexicon, so QAT (D3) *is not possible*.



Position 2-4 Polatnick's Brilliant PEG-8

Best is the peculiar play LEK (4K, 14, EEIOR). Why is LEK better than the obvious move OKE (8A, 23, EEIR)? Many factors contribute equity. First, the vowel/consonant balance of OKE seems better, until you look at the unseen tiles and discover that there are just two vowels left among 15 tiles. LEK's 4-1 leave is actually better than OKE's 3-1 leave. Another difference is that OKE creates a new line for the opponent to play a bingo. Since OKE takes an 81-point lead, it is likely that you will win even if the opponent does bingo, but why court trouble? Besides the general admonition against courting trouble, in this position there is a specific danger: there is a Q in the bag, and there is no U available. Therefore, if you draw the Q after playing OKE and the opponent plays a bingo then you *will* lose. Thus, we see another advantage of LEK: it

plays only two tiles, which reduces the chance of drawing the Q. It appears to reduce the chance from 3/15 to 2/15, but the actual reduction is greater. We know that the opponent did not keep the Q last turn, because he would have exchanged the Q rather than play CAY. Since the 5 tiles the opponent kept were not Q, the true reduction is from 3/10 to 2/10.

MAVEN actually contains these heuristics in some form, so MAVEN ranks LEK close to OKE. LEK has additional things going for it, which are hard to see.

1) Because of the surplus of consonants in the bag, it is valuable to interfere with useable vowels on the board. LEK interferes with a cluster of vowels, whereas OKE opens another vowel.

2) JURA (L1) plays for 44, so we take away a play that could help the opponent to win. OKE blocks JAR (8A), but that scores only 32. The extra 12 points make a difference when we are stuck with the Q.

3) Here is the kicker: LEK sets up QUEER (L1, 29) if we draw the Q, which eliminates the stuck-with Q scenario. Therefore, the game is impossible to lose, unless the opponent somehow blocks (or plays) QUEER, and then bingos. That scenario is unlikely, because there are few tiles in the bag.

How would you write computer programs to see these things? It is hard to generate such insights, and hard to evaluate them. When this problem was presented to a panel of 24 experts [9], only one (Steven Polatnick) found the best move. Scrabble choice-of-move decisions can be difficult indeed. Despite the difficulty, human masters regularly have such insights in the pre-endgame. This document gives other examples elsewhere.

2.4.2 Variance

Another possibility is that humans can manipulate variance to their advantage, with the idea that they can lose games by large margins, provided that they win more games by close margins. Examination of the table of average scores should convince you that variance is hard to manipulate in the early game. Positions show relatively uniform variance through the first half of the game, whereupon the variance decreases towards the end. This is a natural consequence of the size of the board; as the tile configuration expands, new hotspots arise for every hotspot consumed, but eventually that dynamic peters out. During the expanding phase of the game, a player can do little to affect variance.

Moreover, because winning chances in a long game are approximated by a cumulative normal distribution function (a consequence of the Central Limit Theorem [10]), the winning chance is determined by the lead and variance using a function of the form Lead / √Variance. Accordingly, the variance has a second-order impact on winning chances. The average score has to fall by a lot before changes in the variance are large enough to outweigh changes in score.

The author's conclusion is that manipulating the variance is generally a useful tactic only towards the end of the game. At the end of the game, the Central Limit Theorem no

longer applies, and the Variance of the next turn is more significant relative to future turns.

2.4.3 Generalities

The general shape of a computer-human game, then, is that the program will have the edge until near the end of the game, whereupon it is up to the human to snatch victory from the jaws of defeat. Scoring dominates the early part of the game, so computers should be adept at it. This assumes, of course, that computer programs can do a reasonable job of evaluating which tiles to keep. Computers will misevaluate moves by a few points, but these will be infrequent and of little consequence. More frequent and more significant will be human move generation errors.

The late stages of the game, however, are up for grabs. As the bag empties, the human ability to calculate specific variations becomes more valuable. Therefore, it is increasingly likely that humans will select moves that have higher point differentials.

In addition, as the game shortens, the ability of a human expert to manipulate the variance of the game becomes significant. In the early game, this is not important because the normal approximation of the Central Limit Theorem is a good approximation to winning chances. However, if the number of moves remaining is so small that the normal approximation is not valid, then variance can be a significant factor.

2.5 Plan of Attack

These insights into the space and into computer and human play suggest the following plan of attack.

1) Make a program that generates all legal moves.
2) Add the ability to evaluate racks.
3) Improve the play of the program in the endgame.
4) Improve the play of the program in the pre-endgame.

If the analysis of this chapter is correct, then stage 2 should be approximately at the level of the best human players. Therefore, stages 3 and 4 should put the program in the superhuman category.

The remainder of this thesis will show that this plan basically succeeded. However, at the start of the project it was far from clear that success was assured, even though the author was overflowing with confidence.

Chapter 3 – Brief History of MAVEN

The history of computer Scrabble is poorly documented, which the author recognizes as partly his fault. The goal of this chapter is to make amends. The chapter covers factors that influenced MAVEN's development, in historical order.

The theme of this chapter is that human and computer expertise coevolved. MAVEN's early development established a statistical basis for positional play in Scrabble, which trumped a logical, but unsound, human theory. There followed an "arms race" between MAVEN and human masters, in which MAVEN's tactics were refined to match the best of human practice, while humans adopted many of MAVEN's methods.

3.1 The Early Literature

Around 1980 the question arose as to whether computer programs were able to play competitive Scrabble. In 1982, the first attempts were published [13,14].

These early programs were weak. They were constrained by small memory sizes, as they ran on the microcomputers of the day. Because these computers were slow as well, they limited themselves to selective search of hotspots. Truly, it is remarkable that they were able to average even 20 points per move.

3.2 My First Experience

The author read the early literature in the summer of 1983. Surpassing the published results seemed feasible. I had heard that human experts averaged 30 points per move, so there was a lot of room on the upside. Moreover, the limited vocabulary and selective search of the early programs seemed like obvious targets for improvement.

My first program was written in PL/1 on an IBM mainframe in the summer of 1983. The program generated all legal moves using a fixed word list, which contained about 25,000 words out of the Official Scrabble Player's Dictionary (OSPD) [15]. I typed in all the short words and all JQXZ words, as my instinct was that these were the most important words. The contents of a spell-checking dictionary supplemented this list, which gave the program an occasional bingo.

This was sufficient to significantly surpass the programs described in the early literature. The program averaged about 23 points per move. It also yielded a glimpse of the future, as it averaged over 30 points per move in one game. It seemed that the advantage over the earliest programs was attributable to better move generation (both larger vocabulary and exhaustive generation rather than selective), since there was no improvement on the strategic side.

The program's rack was often clogged with multiple I's and U's. Of course, this is symptomatic of a program that always chooses the highest scoring move: the tiles that tend to occur in high-scoring moves are played off, leaving tiles that are less playable. Several turns of this dynamic can result in an accumulation of such *drek,*[11] especially if there are few words that play multiple instances of bad tiles.

[11] A colorful term used by players to describe bad racks.

The author wondered what would happen if the program had a bias towards keeping S and blank and playing away duplicates. To avoid interfering with the strategy of choosing the highest scoring move, this bias became a tiebreaker to apply when moves had equal score. To my delight, the program scored almost 25 points per turn with this modification. It seemed that rack management was another opportunity.

3.3 First Commercial Software

The middle 1980's saw the release of early commercial efforts. Alas, nothing from the author! A clever entrepreneur obtained worldwide rights to Scrabble computer software from the copyright holders, who basically gave away the farm for a period of 15 years.

The commercial software (published by Virgin Games) was a financial success, but a catastrophe for AI, in the author's opinion. The programmers were not at fault, because the positioning of the products required that they operate on computers that had extremely limited amounts of RAM. Accordingly, these programs had deficient vocabularies and ran slowly.

Experts were familiar with a handheld game machine named MONTE PLAYS SCRABBLE. MONTE was introduced with great fanfare, including a demonstration match against a human expert. I recall that the human, who was expert but not of championship caliber, won 50 games without a loss. The experience convinced human experts that computer programs could not compete.

Of course, poor vocabulary doomed the early programs from the start. Human experts also criticized their strategy, but this criticism is misplaced. If a program scores only 20 points per move then strategy is simply not an issue. No strategy will overcome a deficient vocabulary.

However, human experts took the strategic weakness of these early programs as evidence that Scrabble was too hard for computers. They believed that expertise required human *intuition*, and considered human champions invulnerable.

3.4 The First MAVEN

MAVEN's first incarnation debuted in 1986. It ran on a VAX 11/780 and was written in C. It contained the full OSPD dictionary (well, almost) and a rack evaluation function that was tuned by self-play. It was named MAVEN, a word of Yiddish origin, because the OSPD defines MAVEN as "expert". Later on, someone who actually knows Yiddish told me that the connotation is more "know-it-all" than "expert." So much the better!

MAVEN performed many self-play experiments. These showed that MAVEN averaged 34.5 points per turn, which was higher than the standards of human experts. At that time the program was, in the author's opinion, already the world's best player.

The rack evaluator was a great success. It routinely played reasonable rack management strategies, such as keeping Q and U together, saving blanks for bingos, and splitting duplicates. It showed great ingenuity in saving better tiles when it had the option. For example, on an opening turn it would play BLOCK rather than BLACK, because keeping an A is better than keeping an O.

It was fascinating to see how the simple tradeoffs in the rack evaluator combined to produce intuitive solutions to complex problems. When two moves that had little in common were compared, MAVEN was adept at choosing what seemed (to me) to be the more promising alternative.

The program generally showed great strength in the early and middle game, where many moves were forced for scoring reasons. However, there were some clear glitches. For example, it opened triple word squares too readily. It had difficulty with vowel/consonant balance, in that it would rate a rack like ANST as worse than NRST because R is generally better than A. However, if you already have 3 other consonants then surely the reverse is true. Nevertheless, the program seemed fantastically strong overall in the early and middle game.

In the endgame, however, the program made obviously shortsighted plays. The general rack evaluation concept was too imprecise to guide play in the endgame, which is a deterministic game. Some adjustments to rack evaluation slightly improved matters, but MAVEN obviously needed a search engine. I implemented a rudimentary endgame search engine that selectively searched three plies.[12] It would be able to avoid some errors.

The author met with Steven Root and Michael Wolfberg, two Boston-area experts who had some experience with computer programs. They played a game against MAVEN, which MAVEN won resoundingly. They were impressed by MAVEN's overall play, and made several suggestions for how it might be improved. When asked if they thought MAVEN was championship caliber, they politely doubted that it was, even though it had just drubbed them while playing an entire game of reasonable moves. They asserted that there existed experts far superior to them, and from the reverence with which they spoke of Ron Tiekert and Joe Edley, the author envisioned invincible Gods of Scrabble. I wondered how much better any player could be, since Root and Wolfberg had missed little in their game against MAVEN. It turned out that "far superior" was not as great a difference as one might imagine.

Root and Wolfberg were using a program that was distributed on a Digital Equipment Corporation User Group (DECUS) tape.[13] They set up a game between MAVEN and the DECUS program, but we only made a few moves before our Internet connection dropped. The DECUS program had a complete dictionary and played a reasonable game. It made some dubious moves, so it might not have been as well tuned as MAVEN. Nevertheless, it should have been close, since it was an exhaustive move generator. Since Root and Wolfberg played it frequently, they should know whether it was championship caliber; I figured that MAVEN could not be much better.

3.5 First Tournament

MAVEN entered in its first Scrabble tournament in December 1986. The tournament organizer was Alan Frank, the author of TYLER. Frank wanted to test TYLER in a

[12] A ply of search consists of a move by one side. Thus, three plies would look ahead through our move, the opponent's reply, and our reply to that.

[13] The program was an early version of CRAB, originally developed by Appel and Jacobson. A later version of CRAB appeared in the Computer Olympiads, with modifications by Scrabble experts Steven and Graeme Thomas.

tournament against top experts, so he arranged a tournament for this purpose. Frank was congenial about allowing another program into the act.

I talked to a few players at the start of the event. The participants were polite, but skeptical about the program's chances. Or doubtful, or downright dismissive. Forrestt Tellis described MONTE and Virgin's program and averred there was little chance that any program could stand up to real experts. He allowed that maybe TYLER would be able play well, since Frank was a master player, but I should not be surprised if MAVEN did not do well. Tellis is a nice fellow, and he was trying to prepare me for the worst. When told that MAVEN averaged 34.5 points per turn, Tellis said that was a lot, but we would not know what it meant until MAVEN played real experts. You see, it is a self-fulfilling prophecy; MAVEN is unknown, so no statistics of it are knowable. In order to earn respect, MAVEN would just have to kick some real expert butt.

MAVEN's first game was against Alan Frank. MAVEN got off to a good start, and laid down two quick bingos. Then Frank came back with two bingos of his own. MAVEN kept a steady lead through the end of the game, and that is the way it ended. When asked what he thought, Frank gestured to a move and said, "I thought this was a horrible play." Then he got up and left! Frank was top-ten ranked, so this must have been a terrible blow. He was the first player of real standing to lose to a computer program.

In the first round TYLER was kicking expert butt, too. Chris Reslock was one of the top ten players, and at one time was the highest rated player, one of only 10 players to this day who can claim that distinction. TYLER was winning until the last move, when Chris dumped all seven of his tiles to score a gazillion points. The word was phony, but Chris was relying on a programming error: that TYLER would test for end-of-game before testing for whether to challenge. Chris lucked out, and TYLER accepted the play. A double-whammy for Alan Frank in the first round. I scurried to verify that MAVEN implemented the challenge test correctly, which was handy when Edley tried the same ploy on MAVEN in a later round. (SWARTZI, indeed!)

In the second round MAVEN lost to Michael Wolfberg. MAVEN was cruising along when it hit the endgame. It made a play, and Michael suddenly seemed energized. He scanned the board carefully and made his move. With a sinking feeling, I looked at MAVEN's reply; Michael had blocked the only place where MAVEN could play its last tile, a C. MAVEN was stuck with the C while Michael played out one tile at a time, maximizing his total. In the end, MAVEN lost by 8.

MAVEN's endgame code should have prevented that. I could not find any coding errors. MAVEN's endgame code also failed in another game, though it did not cause a loss. I then disabled that section of code. I found the bug after the tournament: I flipped the sign in a minimax operation, with the result that the search engine always chose the same move as the heuristic play would have. MAVEN would have defeated Michael Wolfberg were it not for that bug.

Over the rest of the tournament MAVEN went 7-1, including wins over Robert Felt (National Scrabble Champion, 1990), Joe Edley (National Champion in 1980, 1992, and 2000), Chris Reslock, and twice Jim Neuberger (twice runner-up in the Nationals). MAVEN was impressive in the substance of its wins as well, since it averaged a 70-point per game spread.

MAVEN's other loss in the event came largely because of operator error. Early in a game against Rose Kreiswirth, I neglected to enter Kreiswirth's play, whereupon MAVEN generated a move as if Kreiswirth had passed. The tiles that MAVEN played overlapped the tiles that Kreiswirth had played, creating a phony. I did not notice the discrepancy. Kreiswirth challenged. I appealed to the tournament director that correctable operator errors are not normally charged against programs, but the tournament director disagreed. I still think the decision was wrong. MAVEN lost its turn, and eventually lost by 30 points. There is no guarantee that MAVEN would have won if it had not lost a turn, of course.

However, one of MAVEN's wins was tainted by operator error, so I cannot complain. Robert Felt was winning big when MAVEN laid down 3 consecutive 86-point bingos to take the lead. The words were JOUNCES, JAUNTIER, and OVERTOIL. (Some things you never forget.) Then several moves followed. Late in the game, MAVEN wanted to challenge a word (BA) that had been played several turns before. The tournament director came over, and MAVEN lost its turn, but won anyway because it had a large lead at that point. The win is tainted because had I noticed that MAVEN wanted to challenge when BA was actually played then the course of the game would have been significantly altered. MAVEN still would have had its large lead, but no lead is ever safe.

3.6 Impressions from First Contact

My impression is that MAVEN was better than the humans of the day were, notwithstanding holes in its vocabulary. While MAVEN could make errors because its vocabulary was not 100% accurate, it more than made up for them with superior understanding of rack evaluation.

At the time, humans evaluated racks by relying on a notion of "turnover."[14] They observed that the player that played more tiles was usually the winner. Accordingly, they emphasized moves that played many tiles. This theory confuses cause and effect; surely the player that won had better tiles, and therefore was able to play more of them! Alas, without computer support humans could not statistically validate theories about the game, so they invented models to support the turnover theory.

For instance, the theory went that turning over tiles would improve your odds of drawing a blank, which helps twice. Not only do you draw the blank, but your opponent does *not* draw it. With this fallacy, human expert effectively doubled the value of a blank in the bag relative to one in the rack. The error was in considering a blank in the bag to be worth something to the opponent. In reality, the tiles in the bag will split equally, so a blank in the bag has no net value to either side.

To make the theory concrete, Joel Wapnick recommended [8] that in the early game each tile played is worth 2 to 3 points, with the exceptions that keeping an S is worth 8 points and keeping a blank is worth 40 points. To see the internal inconsistency of this model you need only compute the average value of all of the tiles in the bag. Specifically, 2 blanks make 80 points, and 4 S's make 32 points, whereas 94 other tiles at –2.5 points

[14] Except for Dan Pratt, who had a penchant for setups and fishing and was, according to Joel Wapnick, "way ahead of the rest of us." Ron Tiekert may also have been ahead of the curve, thanks to his pioneering use of simulations.

each make –235 points. Is the average tile in the bag actually worth (80 + 32 – 235) / 100 = –1.23 points? Is it possible for the average of all tile values to be anything but zero?[15]

These inconsistencies in model were insurmountable. MAVEN's opponents considered MAVEN's moves to be *fishing*, a style of play that was deprecated among masters. A fish is a move that accepts a smaller score now in order to increase the chance of a bingo next turn. For example, MAVEN might take 15 points with a 1/3 chance of a bingo next turn. A human considers this to be fishing, and prefers to score 30 points this turn. But he is wrong—if a bingo is 75 points then the average score of a sequence in which you get 15 with chance 2/3 and 75 with chance 1/3 is 35 points per turn. If anything, MAVEN should fish more often.

Human masters observed MAVEN getting great racks on turn after turn. At first, it looked like a mere hot streak, but no streak can last so long. MAVEN dominated both the tournament games, and the informal games. In most informal games, MAVEN played against a team of humans, which reduced their chance of move generation error. MAVEN dominated anyway.

Surprisingly, it seemed that MAVEN had better *positional* understanding than human experts did. Human experts criticized MAVEN's positional play for being too aggressive, but it is likely that human experts of the day actually hurt themselves by overestimating the significance of positional factors. To be sure, MAVEN can place itself at risk with wide-open play. However, it is equally clear that humans of the day hurt themselves by playing timidly.

After repeated thumpings, some players began to wonder what was going on. Robert Felt discussed MAVEN's positional data with the author until the early morning hours. He was disenchanted with his approach to positional evaluation, and thought that some of MAVEN's data had the potential to revolutionize Scrabble.

Peter Morris was something of a maverick among top masters, since he eschewed studying low probability words.[16] Morris was a big advocate of tile turnover, and he saw MAVEN's play as validating his approach to the game.[17] But while MAVEN did turn more tiles than its opponents (by a margin of 55 to 45 in the first tournament) the mechanism it used to achieve that outcome differed from the method Morris was using. Morris would sacrifice points to turn over *more* tiles, whereas MAVEN would sacrifice points to turn over *bad* tiles. This distinction would not be clear until several years passed.

Other players still did not get it. There is a doctrinaire streak in many top players, and one good tournament was not about to overturn the theories that had brought them to the top

[15] Actually, if you value the draw of a tile from the bag at the average of all unseen tiles, then it does not matter whether the tiles average to zero. In effect, the value assigned to a draw from the bag compensates for any non-zero bias in the average tile value. This does not save the human system, however.

[16] At the end of one game he asked, "SUINT? Is that a word?" Other players, most much weaker than he, were amazed that he did not know the anagram of UNITS. "It's an almost useless word," he remarked in reply. Peter is right; SUINT is in the bottom 10% of all words ranked by utility.

[17] But with better vocabulary.

of the game. These players greeted each win with disbelief. They actually expected MAVEN's streak to end.

By the end of the tournament, it was hard to find players who thought they could beat MAVEN. Finally, Joe Edley announced, "I am going to beat this thing. All by myself." He refused help from other players, and he did win.

In addition to its move generation edge, its superior evaluation of racks, and its positional understanding, MAVEN had two practical advantages in over-the-board play. First is that it is never tired. The tournament schedule had 2 games on Friday, 5 on Saturday and 3 on Sunday. The games after lunch on Saturday were grueling. You could sense the energy level in the tournament hall was lower from the amount of talking, the number of errors, and some general lethargy. The players nevertheless played expertly, but long tournament days are a definite advantage for MAVEN. In a modern North American Championship, the schedule is brutal: 7 games on 4 consecutive days, followed by 3 games on the fifth day, making 31 games over a 5 day period. Computer programs would have an additional edge down the stretch and in all afternoon games.

MAVEN's second practical advantage was its speed of play. MAVEN took about 5 seconds to choose its moves, so with operator time included MAVEN's turns averaged about 20 seconds. Compare that with a typical average rate of 100 seconds per turn in a human game. On many turns, the opponent had not even drawn replacement tiles when it was his turn again. Games at this event used with 30-minute clocks instead of the usual 25, but opponents were in time trouble anyway. Rose Kreiswirth remarked, "MAVEN plays at a...good pace." She was putting a good face on the matter; it was obvious how disconcerting it was.

3.7 Further Development of MAVEN

MAVEN's first tournament turned up several defects. First was that the vocabulary was not 100% accurate. MAVEN had lost two turns by playing phonies (not including the operator error against Rose Kreiswirth) and lost a challenge of a valid word. It also played a phony that was not challenged. Validation of a section of MAVEN's word list revealed a 3% rate of missing words and a 1% rate of misspelled words.

Joe Edley played METHADONE, a nine-letter bingo. The OSPD contains no words whose stem is longer than 8 letters, so MAVEN could not have found such a play.

It was uncertain that vocabulary was MAVEN's biggest defect, but the path to improvement was at least clear, so work started on tasks to improve the accuracy of the dictionary.

MAVEN's endgame was a fiasco. A simple bug played a role, but the program would not play perfectly even if that bug were fixed. Some endgame plays at the tournament involved significant tactical lookahead. It was possible to revive the endgame code, but upon reflection, I decided (correctly) to restart from scratch, so I threw out the old engine (a mistake). With the benefit of hindsight, I should have fixed the bugs in the original search engine, so that the new engine would have a basis for comparison.

Rack evaluation needed attention. Human experts confirmed a suspicion that MAVEN did not always play well when vowels and consonants were out of balance. MAVEN received a rack evaluation heuristic that corrected that limitation.

Human experts maintained that MAVEN was too aggressive about opening up the board. Testing showed that they were mistaken, and suggested that the human approach was self-defeating.

In addition to polishing MAVEN's intelligence, I distributed MAVEN to experts who could provide feedback about the program's play. Many early adopters were not of championship caliber, but they were experts, so their feedback about its occasional lapses was valuable.

3.8 Olympiads

In 1989, 1990, and 1991 there were Computer Olympiad events in Scrabble. I had heard about these events, but declined to participate. Participating in the first event was impossible, since I heard about it too late to prepare. MAVEN could have focused on preparing for the second Olympiad, but there were several reasons not to.

1) The Olympiad dictionary was SOWPODS, whereas MAVEN used the OSPD. Typing in a new dictionary had little appeal, especially since MAVEN had no users that played SOWPODS.

2) The Olympiad was too short to distinguish the programs. MAVEN had about a 10-point per game edge over the other programs, owing to its endgame engine, but that margin is not decisive in a short series. For example, the 10-point edge would produce a winning percentage of about 53%. So a 100-game series should end at 53-47, but there is about a 30% chance that MAVEN would actually finish under 50%.

3) I wanted to work on simulation instead.

Therefore, the Olympiads took place without MAVEN's participation, and they do not seem to have suffered from the omission. Table 3-1 shows the participants, who made a worthy demonstration of the state of the art in those years.

Name	Authors	Results
TSP	Homan	1st in 1990 and 1991
CRAB	Appel, Jacobson, Thomas and Thomas	1st 1989, 2nd 1990
TYLER	Frank	2nd 1989 and 1991, 3rd 1990
QUETZAL	Hooker and Guilfoyle	Competed in 1989 and 1990

Table 3-1 Olympiad Participants

CRAB, TYLER, and TSP had indistinguishable results, with no more than 3 games separating first from third place in any event.

Because MAVEN never entered an Olympiad, there is little experience in direct comparison of MAVEN against other programs. There are occasional rumors that some program or other is better than MAVEN, but I do not take such claims seriously. MAVEN

has no automated mechanism for playing against any opponent, so any head-to-head comparisons are necessarily short. Moreover, the only public copies of MAVEN are at least 7 years old, and were tuned to OSPD1 and then upgraded to TWL98 without retuning. Anyhow, I have always regarded that other Scrabble programs were equal to MAVEN except for endgame skill, so competitions against other programs seemed pointless. Would you believe that I have witnessed only 11 games played by MAVEN against other computer programs? It is true.

The 11 games mentioned above occurred in 1989 because TYLER and MAVEN were "seat racing" for the right to sit on first board in a "team tournament."[18] MAVEN won the first game handily, but then Alan Frank found that TYLER was set to play at its default skill level rather than its top skill level, so we threw that game out. MAVEN and TYLER then split the other 10 games evenly. TYLER had higher point differential, so it played first board for the computer team, and MAVEN played second board.

Unfortunately, I lacked awareness that history was being made, and did not preserve the game scores. Therefore, I have no actual competitive games to show you. Instead, I will annotate a game from a Computer Olympiad. The course of this game will show other programs playing at their best, and will represent how MAVEN played in 1986. The annotations also reveal much about how the 2002 edition of MAVEN differs.

This game is from the Second Computer Olympiad. Computer Olympiads use the SOWPODS dictionary, which consists of a merger of North America's OSPD and the United Kingdom's OSW. The same dictionary is used in the World Championship, and in regular play in much of the world. The merged dictionary exceeds its constituents by about 50%, making the game a sterner test of skill than when using either dictionary alone.

It seems that the Olympiads had recurring hitches involving vocabulary and challenge adjudication. For instance, in the 1989 event, none of the programs had reliable vocabularies. In the 1990 competition, QUETZAL played 64 out of 96 games without part of its vocabulary, but fixed the bug for the final 32 games. Because games were played in matches of 16, QUETZAL's last two opponents had a harder time than QUETZAL's first four opponents did. In the 1990 competition, TSP lost when it played GAEN, which an opponent challenged. GAEN is a valid North American word, but was incorrectly omitted from an early printing of the OSPD. The Olympiad tournament director did not possess an up-to-date listing of corrections to the OSPD, and ruled incorrectly. In the 1991 competition, TSP was unaware that Webster's Ninth Collegiate Dictionary was the reference dictionary for nine-letter words, and failed to include such words in its vocabulary. TSP actually challenged two nine-letter words played by TYLER, but the tournament director did not have a copy of Webster's Ninth at the tournament hall, and allowed TSP to escape without losing a turn [47].

TSP seems the equivalent of 1986 MAVEN, consisting of a move generator and rack evaluator with a more-or-less complete dictionary. Its author wrote [12],

[18] In a team tournament the players are grouped into teams that play matches. Similar to the organization of chess Olympiads.

> "Surprisingly, TSP won the competition despite employing less strategy than the other programs. TSP uses rack management, but pays no attention to defense and has only a very crude endgame strategy."

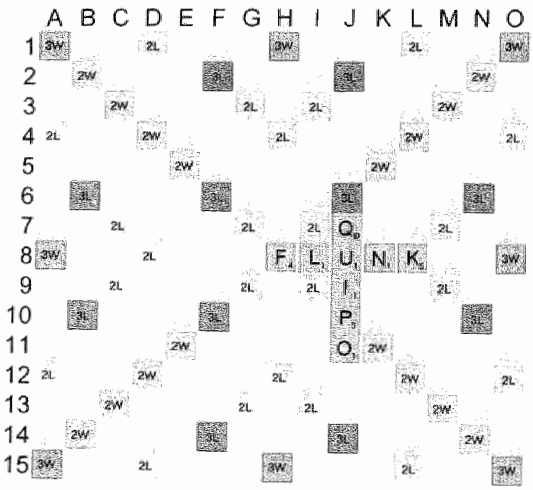
Chapter 6 will show that the fact that TSP won using less strategy is *not* surprising. Nevertheless, it would have been surprising to the observers of the day.

CRAB's racks have not been preserved (it seems I am not the only Scrabble programmer lacking a sense of history), so I am constrained to annotating TSP's moves.

CRAB's first move was to exchange tiles, so TSP played to an empty board with the tiles FIKLNTU. TSP's move FLUNK (8H, 34, IT) is obviously best. CRAB responded by playing QUIPO (J7, 22) through the U in FLUNK to bring up the position at right.

TSP played AWEE (K10, 28, IIT), which has the obvious drawback of keeping II. Now I suspect that TSP was not penalizing the retention of II enough, because WAIT (K11, 24, EEI) seems better from the perspective of score + rack leave. Nevertheless, TSP found what may have been the best move. Nowadays we would recommend WAIN (K5, 25, AEEIT), but I recall that QI was unacceptable at the time.

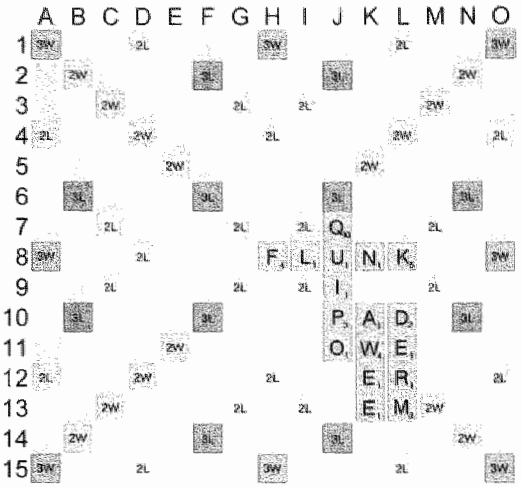
The phenomenon of finding a better play because your evaluation parameters are off is actually typical. Rack evaluation parameters represent only an average over all positions, and therefore might not be representative of this position.



TSP:	A E E I I T W	34
CRAB's last:	QUIPO (J7, 22)	22

After CRAB's QUIPO (J7, 22)

CRAB played DERM (L10, 34), overlapping AWEE, to bring up the position at right. TSP played OUSTITI (7B, 63), which is the only playable bingo. One characteristic of Scrabble is that it is difficult to tell which racks have bingos. TSP found a bingo in this scrap heap of a rack, yet in the promising rack ADEINOT you will find nothing.

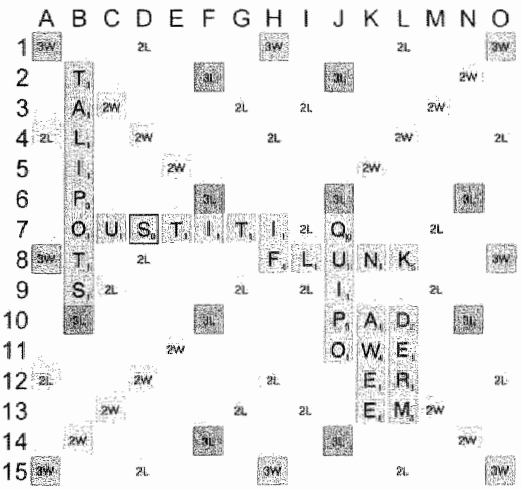


TSP: _ I I O T T U 62
CRAB's last: DERM (L10, 34) 56

After CRAB's DERM (L10, 34)

After CRAB's TALIPOTS (B2, 82), which was CRAB's only playable bingo, TSP found the beautiful double-double BOTCHED (E5, 60). Note the contribution of the premium squares. Racks laden with heavy consonants can be unproductive if the premium squares do not cooperate, or they can be huge. CRAB replied with the clever extension FLUNKEYS (8H, 54), adding EYS to FLUNK to hit a triple word square.

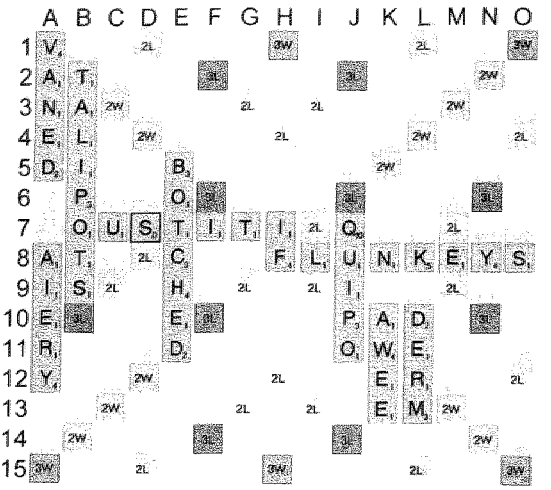
TSP then held ADENRVV. TSP played VANED (A1, 40, RV), which is clearly best. The alternative is the fish VAV (3A, 18, ADENR), which has a high bingo potential. The average score of TSP's turn after playing VAV would be 63 points. Perhaps VAV would be the best play if TSP needed a bingo to catch up.



TSP: B C D E H N O 125
CRAB's last: TALIPOTS (B2, 82) 138

After CRAB's TALIPOTS (B2, 82)

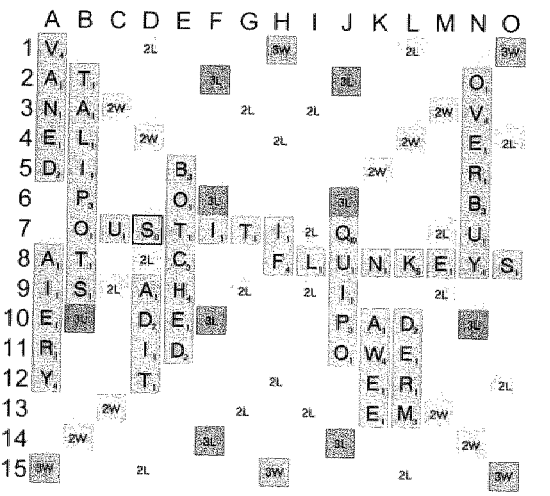
CRAB played AIERY (A8, 44). TSP found the nice move OVERBUY (N2, 42), which is clearly best. Nothing else comes close. Note the accuracy of the programs when the moves are obvious. Also, note that when the moves are not obvious, the programs choose something reasonable.



TSP: B, E, N, O, R, U, V, 225
CRAB's last: AIERY (A8, 44) 236

After CRAB's AIERY (A8, 44)

CRAB played ADIT (D9, 21), and now TSP played a move that is just plain wrong. TSP played CENT (C1, 24, GIO), which keeps terrible tiles. MAVEN rates INCOGS (D2, 23, ET) as 14 points better than CENT. Is it possible that OB was not in SOWPODS in 1990? If not, then why not play COIGNE (4I, 18, ET), which is also obviously better than CENT? My conclusion is that TSP's rack evaluator was off.



TSP: C, E, G, I, N, O, T, 267
CRAB's last: ADIT (D9, 21) 257

After CRAB's ADIT (D9, 21)

After CRAB's LANAI (14K, 26), a pretty play that hooked an L onto AWEE and an A onto DERM, TSP held GGINORW. TSP played CROWING (1C, 42, G), which was obviously best. In such situations, weaker humans (including the author) often go into a funk contemplating the bingo sitting on their rack that has no place to fit on the board. "If I had an N I would have WRONGING, or an L would make GROWLING. But *of course* neither one plays, and neither does GROWING, ..." If luck has been very bad, then the train of thought can go for several minutes.

CRAB played AXE (F9, 63) to reclaim the lead. The normal move for TSP in this position is FUGIO (O11, 30, HIS), and I cannot see any reason not to play it. FUNG (M12, 32, HIOS) is plausible, but one should prefer FUGIO based on rack leave alone. When you factor in the benefit of blocking the O-column, which is important because of the unseen Z, then FUGIO stands out. MAVEN's simulations give FUGIO an edge of 12 points over FUNG.

TSP:	F, G, H, I, O, S, U,	333
CRAB's last:	AXE (F9, 63)	346
15 Unseen:	?AEEGJLMNOORRSZ	

After CRAB's AXE (F9, 63)

TSP's preference for FUNG over FUGIO is revealing. It shows that TSP had no triple word evaluator, since the O12-O15 opening would be worth 3 points at a minimum, and I cannot imagine that TSP would rate FUGIO more than 3 points behind FUNG. After all, FUGIO only scores 2 points less, and the O it keeps must be at least a slight negative. Another inference is that TSP's rack value for O is greater than –2 points. Or, we might infer that TSP's vowel-consonant bonus was inappropriately weighted to prefer 2-2 distributions over 1-2 distributions. Careful analysis of AWEE and CENT could yield additional information.

I believe that an analysis of the moves of a handful of Scrabble games played by a computer program would produce accurate estimates for the rack evaluation parameters employed by that program. This information would form an opponent model for use in simulations, with the result of specifically defeating that program.

The worst happened when CRAB played ZEIN (O12, 69), to take a 50-point lead. To come back from such a deficit, TSP needs to play a bingo. Since there is no bingo in AHILOOS on this board, TSP's problem is to play off the tiles that maximize the chance of a bingo next turn.

TSP: A, H, I, L, O, O, S 365
CRAB's last: ZEIN (O12, 69) 415
9 Unseen Tiles: ?EGJMORRS

After CRAB's ZEIN (O12, 69)

TSP played OOH (6D, 28, AILOS). OOH is a routine move, following normal midgame principles. Unfortunately, OOH empties the bag, which allows the opponent to block bingos that might come out of the bag. In such situations, it normally pays to leave one tile in the bag. For example, OH (6E, 26, AILOOS), unfortunately, this leaves a duplicate O on the rack, and reduces the chance of drawing a blank out of the bag. On balance, emptying the bag is not as bad as it normally is, because the blank gives TSP a reason to draw more tiles out of the bag. But there is a more accurate move: AH (O1, 20, ILOOS), which wins about 16% against OOH's 10%.

At last, we know CRAB's rack, because all of the tiles are out of the bag. The tournament book noted [12],

> "An analysis of the game shows that CRAB, leading by 22 points with no tiles left to be drawn, could have won by choosing any of a number of different moves on its last play. TSP's only chance of winning was to play GASOLIER for 74 points at 4H or GIRASOLE at 4G for 70 points. No other play by TSP could get more than 25 points. CRAB needed only to block the area around 4G-4O while scoring reasonably well to sew up the game. A large number of plays would have accomplished this including:
>
> 4J OJIMES 30 points
> 4K JAMES 28 points

CRAB: E, J, M, O, R, S 415
TSP's tiles: A, G, I, L, O, R, S 393

After TSP's OOH (6D, 28)

4I REJONES 28 points
5J MAJORS 28 points
G3 JOIST 26 points"

Note that the commentary above does not state which play is best. Without using MAVEN's endgame analyzer there is no way to know which play was best. The best play was RESTEM (4J, 20, JO). TSP's best response would be ISOLOG (2J, 9, AR), which is needed to block JO (O1, 29). Then CRAB would play out with JOB (6L, 12+4), for a net of 27. JURORS (5J, 24, EM) is as good as RESTEM, after TSP's SIGLA (C11, 19, OR) and CRAB's EME (4L, 18+4). The best of the moves given above is MAJORS, which has a net of 25 after RIOJA (L2, 24, GLS) and RE (2E, 13+8).

Instead, CRAB played JO (O1, 29, ?EMRS). This shows that CRAB did not modify its rack evaluation parameters in endgame situations. Better endgame play results from assuming that each tile left on the rack will be collected by the opponent. This produces correct play for the last two moves in the endgame (i.e., going out and the preceding turn), and the accuracy is pretty good before then, too.

After JO, TSP won the game with GASOLIER (4H, 74+12).

To quote from the tournament report [12],

> "An adversary search to analyze the game tree would find one of the winning sequence of moves. But looking at some numbers from this example points out how time-consuming a complete endgame analysis could be in some cases. With seven tiles including a blank, CRAB had 4,137 possible moves to choose from. TSP also had seven promising tiles left and would have had about 800-900 possible moves in response to CRAB's move."

This game shows the state of the art around the year 1990. We have seen that the programs usually chose the best plays. There were exceptions involving close decisions, but if a move was obvious then the programs never missed. Then in the endgame one program completely missed the point and threw away a game that it should have won.

3.9 MAVEN's Analytic Skill Evolves

By 1988, an intensive review of MAVEN's word lists had almost eliminated vocabulary errors. Endgame analysis then became the highest priority.

MAVEN's first (1986) endgame analyzer failed because of a simple bug. With that fault corrected, the search engine resulted in a clear improvement in play. However, a careful domain analysis showed that the results would be far short of perfection.

Examples from Wapnick's book [8] contributed to the domain analysis, and I slowly gained appreciation for the complexity of the endgames. MAVEN's first truly successful endgame player dates from late 1988. The engine made several sparkling plays, and contributed insights into the post-mortem analysis of other games. Human masters were excited by the ability of a computer program to analyze endgames, since that had always been a weakness of computer Scrabble programs.

My second engine was successful, but still not perfect, as careful analysis showed. MAVEN spent much of 1989 working on endgame analysis. The result was a novel application of the B* search algorithm, which set a standard for endgame play that was not equaled by human or computer for at least a decade. That implementation of the endgame analyzer has proven so robust that the last 13 years have uncovered only a handful of bugs.

A monograph published in 1990 entitled *MAVEN's Endgame Ability—a Comparative Analysis* [16] compared MAVEN's analysis of 54 endgame instances with the moves of human experts. About two dozen experts reviewed this document. Before this publication, no one knew how many errors humans committed in endgames. The examples showed that MAVEN was remarkably accurate even in specially concocted endgame situations.

With endgame play practically perfect by the end of 1990, MAVEN's biggest opportunity lay in the pre-endgame phase. Wapnick's book contained many pre-endgame instances, and MAVEN generated others. I developed a domain analysis of the pre-endgame based upon these examples, and then developed an algorithm that seemed to have good properties. Implementing that algorithm accurately was harder than expected, and it remained rife with bugs for the next 7 years. Notwithstanding the bugs, from 1990 onward MAVEN had a pre-endgame module that improved upon standard computer play.

3.10 MAVEN and Simulation

A simulation is a Monte Carlo search of the positions near a move-selection point. The goal of simulation is to estimate the point differentials of the moves that you might make, under the theory that the move with the highest point differential is probably best.

MAVEN's simulation capability dates to 1989. At first, simulations took so long that they ran overnight. However, simulations became faster as computers became faster, and it was not long before you could complete a simulation in less than 30 minutes.

Simulations revealed a wealth of information about Scrabble. MAVEN's human colleagues regularly published amazing MAVEN simulations. For example, simulations revealed that on the first turn it was generally better to play a short word, other things being equal. This was exactly the opposite of the conventional wisdom.

MAVEN's ability to perform simulations made it an essential tool for experts who were serious about positional theory. Several experts even purchased Macintosh computers in order to run MAVEN, since at the time MAVEN was not available on PCs.

By 1996, computers were sufficiently fast that MAVEN could use simulations to select moves in real time. By 1998, MAVEN's implementation of simulation was robust enough that it was a huge advantage in practical games.

3.11 MAVEN's Evolution and Human Expertise

Human expert play has gotten considerably more sophisticated since 1986, and in the author's opinion, MAVEN has driven much of the progress. It is interesting to compare MAVEN's progress with human progress over the period from 1986 and 1997.

At first MAVEN's advantage was attributable to accurate move generation (in principle, it considered all possible moves) and fine-tuned rack evaluation. However, my constant preaching about the advantages of MAVEN's rack evaluation steadily eroded that advantage. By the early 1990's, several experts had adopted MAVEN's theories. Their successes attracted the attention of other experts. Table 3-2 shows how talented MAVEN's early adopters were.

Player	Peak Achievements
Peter Morris	1989 National Champion, 1991 World Champion
Robert Felt	1990 National Champion
Charlie Carroll	1991 Masters Champion
Adam Logan	1996 National Champion
Joel Sherman	1997 World Champion
Matt Graham	1997 World runner-up
Joel Wapnick	1999 World Champion. 1993 and 2001 World runner-up. 1992 National runner-up.

Table 3-2 Remarkable Achievements of Early Adopters of MAVEN

One particularly industrious expert was Nick Ballard, the publisher of the *Medleys* newsletter. Ballard used cleverly designed simulations to determine the value of each tile and every vowel/consonant balance, which he published in *Medleys* [17]. Ballard also exposed the fallacies of the turnover theory. Many players learned the theory of rack evaluation by reading *Medleys*.

The historical progression of human expertise started with a transition from the crude turnover theory to MAVEN's statistically based theory of rack evaluation. Even players such as Joe Edley, who shy away from quoting numerical values for tiles, have incorporated MAVEN's theory into their repertoire somehow. For instance, Edley quotes [18] the order of tile quality as Blank, S, E, X, Z, R, A, H, N, C, D, M, T, I, J, K, L, P, O, Y, F, B, G, W, U, V, Q. This is exactly the order that MAVEN gives, except that MAVEN places the J lower than Edley does. Why the exception? It is my fault. Edley asked what the tile values were, and I mistakenly quoted –1.5 for the J instead of the correct –2.5. I did not realize that I have been quoting the wrong value until I prepared this thesis.

MAVEN's demonstration that humans make many endgame errors changed the play of experts. For example, Charlie Carroll allocated his time so that 10 full minutes remained for the last few turns. Players now specifically practice endgames. Players have techniques that search for sequences that go out in two moves. Top humans still cannot play endgames as well as MAVEN, but they have taken a definite step up.

The lessons of simulations are perhaps the most far-reaching lessons of all. Every master has learned something from simulations, even if he is not himself a MAVEN user. Simulation results have become the gold standard of positional analysis. In this respect, simulation has the same standing in Scrabble that rollouts have in backgammon.

The trend started with my circulation of a few fascinating results, by word of mouth. Then Nick Ballard published many simulation results in *Medleys*. Soon experts such as Robert Felt and Charlie Carroll were checking every game they played using MAVEN

simulations. Carroll, in particular, was very generous in sharing his insights in the pages of *Medleys*.

The trend towards relying on simulation received official sanction when Joe Edley required that computers check all games published in *Scrabble Players News*. Soon the trend was sweeping the Scrabble world. By the end of the 1990's, you could not publish Scrabble analysis without simulation results because someone would surely ask for them. The trend may have gone overboard, to the extent that simulation has become a substitute for thinking in certain circles. In particular, simulation does not always result in statistically significant conclusions, and simulation does not address all issues.

However, while players may go overboard in using simulations, I can say with confidence that human experts have learned much from them. In the years between 1990 (when simulation became available as an analytical tool) and 1996 (when simulations were first used in competitive play) humans improved their positional skills by studying simulation results. However, it may be doubted that the improvement ever compensated for MAVEN's advantages in move generation and endgame play.

Human improvement notwithstanding, the author believes that MAVEN has maintained at least a slight superiority over human experts since its debut in 1986. This may be the earliest time at which a computer program achieved world-class status over human masters in a non-trivial game of skill. With the advent of competitive play using simulated games, MAVEN is now out of reach of human experts. No human will ever challenge MAVEN on equal terms. In brief: a program that plays almost perfectly is technically feasible at this point, and MAVEN is close.

3.12 Commercialization of MAVEN

Another aspect of MAVEN was its commercialization. MAVEN became part of Hasbro's Scrabble CD-ROM in 1995. In the author's opinion (admittedly biased), it was a huge success for AI. The commercialization of the technology has spread the news that computer programs can compete in Scrabble at the highest levels of play. By 2001, well over 1 million copies have shipped. Many hundreds of thousands of people have played against MAVEN using this product. The author hopes that MAVEN has brought new players into the game of Scrabble, and increased the understanding of those who already love the game.

3.13 MAVEN's Future

The author has produced new versions of MAVEN for Hasbro as needed, but most of the development is now on issues that do not affect the peak strength of the AI. We consider that MAVEN's "endgame" has been reached.

Chapter 4 – Move Generation

As noted in the domain analysis, move generation is the most critical skill in Scrabble, and in contrast to many games, it is a challenging technical problem. We wish to generate all legal moves using simple, fast, and flexible algorithms.

One of the crucial elements of move generation is to have a complete list of legal words. Nowadays such a list is available in electronic form, but this was not always true. The first section lays out the process used to create MAVEN's vocabulary.

Early versions of MAVEN used a constraint satisfaction algorithm to generate moves, as described in the second section. While this algorithm is no longer the best general algorithm, it is of technical and historical interest, and may still be optimal for certain situations. It can be used in conjunction with data structures introduced by the next two algorithms.

In 1988, Appel and Jacobson published a beautiful algorithm that MAVEN adopted instead of its homegrown technique. This algorithm is described in section three. This is MAVEN's algorithm to this day, but there may be an even better choice.

In 1993, Steven Gordon followed up some suggestions from Appel and Jacobson to create the fastest published algorithm, which is summarized in the fourth section. Gordon's dictionary representation uses 5 times the RAM that Appel-Jacobson uses, but it pays off by doubling the speed of move generation.

The fifth section presents an algorithm devised by James Cherry for ACBOT. Cherry's algorithm is based upon permuting the tiles and looking up the resulting word in a dictionary. This is less efficient than Appel and Jacobson's algorithm, but simpler. It is the method of choice when working on computers with extremely limited RAM, such as microcontrollers.

Finally, some important architectural considerations influence the move generator. The final section describes these.

4.1 The Vocabulary

It is vitally important to know all of the words in order to avoid 3 types of errors. First, you do not want to miss a chance to play a word. Second, if your opponent plays a word you do not want to challenge incorrectly and lose a turn. Third, you want to avoid misspellings because you do not want to play a phony word yourself.

We extensively discuss vocabulary because massive knowledge bases are an important component of AI systems. There is a tendency to dismiss problems of scale as mere matters of technique, but this author disagrees with that viewpoint. Intelligent behavior often confronts issues of scale. For example, we have the examples of chess and checkers endgame databases and opening libraries. We have exhaustive solutions of hex, renju, awari, and so on. In natural language processing, we have the example of Word Net. In information retrieval, we have the example of search engines for specialist domains. It is best to have an arsenal of methods for dealing with large knowledge bases.

4.1.1 The OSPD

The early years of tournament play were marked by confusion over which words were good. The rules state only that all players must agree on a dictionary, and English dictionaries differ considerably. The earliest tournaments used several dictionaries in an effort to satisfy all, but this merely resulted in delays when challenges were resolved. Later tournaments standardized on Funk & Wagnalls, but that solution had drawbacks as well.

Merging the vocabularies of the five major dictionaries used by North American tournament players created the Official Scrabble Players Dictionary (OSPD) in 1978. This event standardized tournament vocabulary, making efficient tournament direction a possibility. The OSPD was the ruling authority when the word was 8 letters or shorter. Longer words still had to be looked up in both the OSPD and Webster's Tenth Collegiate Dictionary [46], but such words come up infrequently. Standardizing the vocabulary made meaningful comparison against human players possible, so it was a necessary precondition for achieving a computer champion.

4.1.2 Word List Creation

Computerizing the words was a big job. The OSPD contained about 50,000 main entries, and with inflected forms totaled about 95,000 words, averaging 8 letters long. At the author's typing speed the data entry task would have taken about 4 months of full time effort, and I was *not* working on MAVEN full time. The prospect of delaying gratification for up to a year while typing the book was unappealing.

Fortunately, a technique inspired by the OSPD itself greatly reduces the typing effort. One can write the main entries and then represent the inflected forms as suffixes. For example, a line from MAVEN's original source text might have read

v enter –er –ers

where the "v" means that "enter" is a verb, which implies that "enters," "entered," and "entering" are valid. The –er and –ers notations mean that you can make the noun forms "enterer" and "enterers."

This method reduced the number of entries to 40,000. Additionally, the data could be read off the page, which is easier than mentally composing every word while typing. Typing took about 60 hours.[19]

4.1.3 Postprocessing

Next, a postprocessor transformed the input into actual words. The postprocessor was a small rule-based system, where the rules handled how nouns and verbs take their inflective suffixes as a function of the root word's vowel and consonant pattern.

During the initial quality assurance pass, I worked with the original source form, fixing bugs in the postprocessor. After a few days of bug fixing, it was easier to fix the output

[19] The OSPD has 660 pages, each of which required a little over 5 minutes of typing per page.

than to fix the postprocessor, so I discarded the manually entered text and the postprocessor, which had served their purposes.

4.1.4 Editorial Review

MAVEN's first tournament showed that the vocabulary was deficient. The first pass had omitted about 3% of the entries in the OSPD, and mistyped an additional 1%. Most of the missing words were inflections, so the rate of missing words was not as damaging as it seems. Typos and omissions both tended to occur more often on the longer words, so actually the error rate was not too bad as a practical matter. Still, it is obviously not acceptable to have such an error rate, so the word list needed a thorough editorial review.

The first goal was to absolutely prevent errors among the most important words. The "Cheat Sheet" from the National Scrabble Association (NSA) was invaluable. This invaluable resource occupies just one sheet of paper, and contains all 2 and 3 letter word, all short JQXZ words, and vowel dumps. Preventing errors within these words was important, as the domain analysis shows that such words account for a tremendous fraction of all plays, and their significance is even higher when you add in their participation in hooks and overlaps.

After validating the Cheat Sheet, the author scanned every entry of the OSPD and verified that it occurred in MAVEN's list, and vice versa. This caught a large percentage of the remaining errors. For example, if my editorial review had a 3% rate of omissions distributed independently from the errors of the first pass, then the omission rate after the review would have been 3% of 3% = 1/1111. Such a rate would be barely noticeable in practice, especially if the words tended to be long.

The error rate after the editorial review was about 1/9000, or even lower than expected by chance. Possibly the low error rate is attributable to a technique of searching for errors similar to those already found. For instance, an adjective had been "pluralized" because the entry was marked with "n" (for noun) rather than "a" (for adjective). I searched the dictionary for strings such as "ESTS" and "OUSES," and did find other instances of improper pluralization.

Occasionally a published list of words could be used for validation purposes. For example, consider the list of SATIRE bingos. If a book contained this list, or many pages of similar lists, how would that help validate? The approach of going through such a list one word at a time is expensive, because each word would require an alphabetical search. The prospect of typing such a list is unappealing, especially since it will probably turn up no errors. Moreover, validation should find misspellings as well and omissions, but the approach of looking up words only turned up omissions.

The solution was to write programs that generated the list as formatted in the book. I could then visually compare the generated list with the published list. Most lists are arranged in columns, so it is easy to determine when a word has been omitted, since omissions cause the columns to line up differently. This procedure also detects misspellings that cause words to appear on this list that should not be there.

Several years later, I crosschecked MAVEN's list against the lists of another group that had undertaken the same ordeal. There were exactly 8 errors, so the error rate was significantly lower than expected.

Crosschecking turned up an annoying fact: the OSPD contains errors that are corrected in later printings. The NSA maintains a list of known errors in the OSPD, along with the editions that corrected them. Tracking this list was a new editorial task.

4.1.5 Long Words

Watching Joe Edley find the word METHADONE through separated tiles inspired me to add the 9-letter and longer words to MAVEN. Scanning Merriam-Webster's Tenth Collegiate Dictionary for nine-letter words took 100 hours, which is nearly twice as long as the OSPD took. A few years later the result was crosschecked against the work of another person for the 9-letter words, but the 10-letter and longer words remained rife with errors until 1996, when the NSA developed an official, exhaustive, computerized list. Fortunately, 10-letter words rarely come up, so errors among such words have negligible impact on playing strength.

Nowadays there is an electronic form of the official tournament word list (TWL98) for North America. TWL98 differs from the OSPD by its inclusion of offensive words (ethnic slurs, scatological words, and other barbarisms are legal in tournament play, but have been excluded from the OSPD since 1993) and the long words up to 15 letters. TWL98 also omits definitions, and gives every inflection in full, rather than representing it as a suffix to a stem word. The entire list is alphabetized, whereas the OSPD contains special lists of RE- and UN- words.

4.1.6 SOWPODS

Tournaments outside of North American use a different vocabulary. For instance, in the third Computer Olympiad in Maastricht 1991, two dictionaries were used to decide whether a word was acceptable: the United Kingdom's "Official Scrabble Words" (OSW) and the OSPD. This vocabulary is known as SOWPODS, since Scrabble players cannot look at the letters OSWOSPD without making *some* kind of word out of it.

MAVEN added SOWPODS support when Joel Wapnick was preparing for the inaugural human World Championship. Wapnick contributed his hand-typed list of high frequency OSW words. Later, MAVEN received the OSW in computerized form. (A relief!)

4.1.7 Summary

This example shows how computerization of a knowledge base of reasonable scale (roughly 150,000 words) is made practical by manual and computerized validation methods. There was a lot of labor involved, even with programmatic assistance. Overall, entering the vocabulary took about 160 hours, and validating it took about 200 hours. It was a grueling, mind-numbing task even with all of the tricks. The effort was similar to the work of early chess developers in creating opening books [20].

4.2 Bit Parallel Move Generator

MAVEN's first move generator searched for spots to place words on the board such that the rules of the game were satisfied. Let us take a specific word (QUA) and see how this

might operate. The question is to identify the set of all squares on which the word QUA may start.

Without loss of generality, assume that QUA is horizontal. To play QUA, several constraints must be satisfied simultaneously:

1) An empty square (or the edge) to the left of the Q (since Q is the first letter).
2) An empty square that is adjacent to an occupied square between 0 and 2 squares of the Q (since QUA must connect to the board).
3) Either a Q in our rack or a Q on the board.
4) A U in our rack or lying one square to the right of the Q.
5) Either an A in our rack or an A two squares to the right of the Q.
6) An empty square (or the edge) to the right of the A (since A is the last letter in QUA).
7) The Q, U, and A must not form crosswords or they must make valid crosswords with letters already on the board.

To make it easier to find solutions to such constraints, MAVEN pre-computed the set of all squares that satisfied any such constraints (e.g., the set of all squares that had a U one square to the right). The constraints were represented as bit vectors, using one bit per square. Intersecting a number of these bit vectors identified all squares satisfying all of the corresponding constraints.

This "bit-parallel" method works well on machines with large words (e.g., 64-bits). Its overhead is excessive, however so it is suitable only for whole-board move generation. The other generators are suitable for incremental generation, such as generating setups off of a move.

A crucial efficiency technique exploits the fact that adjacent words in an alphabetically sorted lexicon share a common prefix. The average number of letters that differ between successive words is about 3, whereas the average length of a word is about 9. So you gain a factor of 3 by setting up the dictionary so that shared prefixes are easy to exploit. For example, the DAWG representation of the next section works well.

4.3 Appel-Jacobson Move Generator

In 1988, Appel and Jacobson's algorithm came to my attention [21]. It represents the dictionary as a *directed acyclic word graph* (DAWG) [11]. A DAWG is a reduction of a *letter trie*, which will be described first. Given a list of words you construct a letter trie by subdividing the list according to the initial letter. In Scrabble, you start with 150,000 words, then break that down into 26 lists. You can index the lists by creating a search tree whose root node has 26 pointers, one for each list. If you continue this procedure recursively until every list has zero entries then you have a letter trie. Among the nodes of the trie are many duplicates. For example, the following pattern is repeated thousands of times: "the prefix that you have traversed thus far is a word, and you can add an 'S', but there is no other way to extend this prefix to make a longer word." A DAWG is a trie after all identical subtrees have been reduced. Also, among the 26 pointers in each node many null pointers exist, and it is wasteful to represent them. Instead, a node is represented as a variable-length packet of edges, where only the non-null pointers are represented. A DAWG is isomorphic to the minimal finite-state recognizer for the regular

language represented by the word list. A DAWG is a compact representation of the word list. For example, the 95,000 words in the OSPD take about 750KB when represented as a word list. The DAWG represents the same list using about 275KB. The efficiency comes from "sharing" common prefixes and suffixes over many words.

```
// No programmer can resist naming this routine "WalkTheDawg"
void WalkTheDawg(int NodeIndex) {

    // If the given node is the "dead end node" from
    // which no edges emanate, then there are no words:
    if (NodeIndex == 0) {
        return;
    }

    int edge;

    do {
        // The DAWG is a packed array of edges, where NodeIndex
        // is the first edge for a given node.
        edge = DAWG[NodeIndex++];

        // Extract the character that labels this edge:
        char c = GetChar(edge);

        // If we have that character on the rack:
        if (TileCounts[c]) {

            // Remove the tile from the rack and append it
            // to the word we are developing:
            TileCounts[c]--;
            Word[WordIndex++] = c;

            // If the edge is marked as completing a word
            // then call the handler function:
            if (IsLegalWord(edge)) {
                AnagramHandler(Word, WordIndex);
            }

            // Recursively anagram some more:
            WalkTheDawg(GetNextNode(edge));

            // Restore the tile to the rack:
            TileCounts[c]++;
            WordIndex--;
        }

        // Continue looping until the last edge in this node:
    } while (NOT IsLast(edge));
}
```

Code Sample 1 Anagramming using a DAWG

This dictionary representation enables efficient algorithms. For example, you can search the state machine constrained by the tiles on the rack, tiles on the board, and constraints from overlapping words. It is easy to make a move generator on this basis. Moreover, Appel and Jacobson showed how to use the state machine to generate moves with minimal effort.

Code Sample 1 shows how to use a DAWG for anagramming. The task is to generate all words that can be played using a given rack. The code assumes that the contents of the rack are represented as an array of counters.

WalkTheDawg needs only four features to make a complete move generator: using blanks, handling of tiles on the board, checking for the edge of the board, and handling of crossword constraints. It is easy to add features for the first three, so we will not go into those issues. Crossword constraints are interesting, and the solution introduces a new concept, so we will give the solution in detail.

A crossword constraint is a restriction on the set of letters that can be placed on a square imposed by letters that lie in the adjacent rows. By traversing the DAWG you can determine the set of all tiles that satisfy the constraint (i.e., letters that make a valid word with the crossing letters). The best way to represent that set of letters is as a word of bit flags, using one bit to represent the playability of each letter. Such a word of 26 flags is called a "checkset." Once checksets have been computed, it is a simple matter to test crosswords for validity in WalkDawg.

To summarize, you can make a simple move generator by looping first over all squares, and trying to find all moves that start on that square. The algorithm is WalkDawg, after it has been improved by the addition of handling for blanks, letters on the board, edge limitations, and crosswords.

Though complete, this algorithm is inefficient. In particular, there is a lot of redundant traversing of the DAWG near the start of a word. For instance, suppose that the starting square is 6 squares away from the nearest letter on the board. Then WalkDawg would traverse all combinations of 6 tiles from the rack, at least insofar as the combinations make prefixes of valid words. This is not bad, inasmuch as it is unavoidable. But when we generate the moves one square to the right, then the inefficiency of the algorithm is clear, since it traverses all prefixes consisting of up to 5 tiles from the rack, which is a subset of the prefixes already generated.

Appel-Jacobson eliminated that inefficiency by defining the concept of "spot" in a better way. They introduced the concept of *anchor square*, which is the leftmost square of a move that is adjacent to a square already occupied. With the exception of the first turn and exchanges (which are handled as special cases), every move has an anchor square.

An anchor square is an important concept because the letters to the left of an anchor square are not constrained by the board (except that the left part cannot occupy a square adjacent to an occupied square), whereas those to the right are. Appel and Jacobson exploit this property by generating words in two parts: the left part, which is constrained only by length and tiles, and the right part, which is additionally constrained by the board.

Appel-Jacobson is a fast algorithm. The bit-parallel algorithm is faster in the early game, because the expense of generation is dominated by left-part traversals in that stage, which the bit-parallel algorithm handles in parallel. However, as the board fills up, the bit-parallel algorithm loses this advantage, and eventually its overhead for maintaining the possibility of parallelism is too high.

A comparison of their algorithm against MAVEN's algorithm showed equal speed, but Appel-Jacobson was simpler thanks to the DAWG. Therefore, we replaced the bit-parallel algorithm with their algorithm, and have not switched since.

Appel-Jacobson has another advantage that is important when designing for endgame search engines: Appel-Jacobson can be adapted to generate moves for only a subset of the board. For example, you can generate setups off of the opponent's last play by generating only those anchor squares that can be affected by the squares occupied by the opponent's last play. Performing such an incremental move generation using the bit parallel algorithm would be comparatively expensive.

A CDAWG (Compact DAWG) is a DAWG in which edges are labeled with character strings rather than only single characters [35]. A CDAWG potentially reduces space and time by identifying forced sequences of transitions that the program can follow in an inline loop rather than by a recursive call. However, in tests using MAVEN, CDAWGs were unable to match the speed of DAWGs. My assessment is that the CDAWG representation is efficient for relatively sparse state machines, but suffers from overhead on dense state machines.

4.4 GADDAG Move Generator

The largest inefficiency in Appel-Jacobson is that the left-parts are almost the same for all anchor squares. Moreover, it is expensive to enumerate left parts because they are constrained only by length and tiles. In effect, at each anchor square one generates all possible left parts, then check to see whether any of them can be extended to be compatible with the constraints of the right part. Appel and Jacobson speculated that it might be possible to build a faster algorithm if you could generate the right parts first, and then generate only the left parts that were compatible.

Following a suggestion of Appel and Jacobson, Steven Gordon (mathematician and master player) implemented a move generator that uses a "two-way DAWG," dubbed a GADDAG [22], which is really a new lexicon rather than a new data structure. In the GADDAG, you represent the word FADGE as the following list of words:

FADGE#, ADGE#F, DGE#FA, GE#FAD, E#FADG, #FADGE,

where the symbol # means "end-of-word." A GADDAG is a DAWG that contains all such words. Generating moves using the GADDAG is simple: you traverse the GADDAG starting at the anchor square and moving to the right. When you hit the # you switch to traversing the left part.

People have experimented with different methods of representing the left parts so as to minimize the cost of traversing them. Some programmers list the left parts in reverse order (e.g., instead of GE#FAD, the entry would be GE#DAF). Tests show small differences in performance and GADDAG size depending on this decision, but the differences are so small that they could be due to irrelevant factors. Perhaps there are significantly better ways to order the left parts. The author speculates that ordering the letters in order of decreasing frequency may be better, so that instead of GE#FAD the GADDAG contains GE#FDA. This ordering would rapidly eliminate left parts that do

not match the rack. Unfortunately, you would have to augment the GADDAG with information about how to reorder the left part so as to reconstitute the word.

MAVEN did not switch to Gordon's algorithm even though this algorithm is twice as fast as the method MAVEN uses. The reason is that the GADDAG uses 5 times as many bytes as the DAWG, and that was too much RAM. MAVEN was designed to run on computers having only 1 MB of RAM, where the GADDAG alone used 2.5 MB. With some regret, I declined to implement the GADDAG, though I should probably reverse this decision, with RAM being in abundant supply nowadays.

4.5 Permutation Move Generator

James Cherry devised a move generator for ACBOT based upon permuting the letters in the rack. Cherry's algorithm is not as fast as Appel-Jacobson but it can be implemented on small-memory computers. For example, we implemented it once on an 8-bit microprocessor using only 96 bytes of RAM.

The idea is to loop over all starting squares for words, and then generate all permutations of our tiles. Lay those tiles down in the empty squares starting at that point and see whether they make a word. In order to make this algorithm fast you need to determine the leftmost prefix of your word that does not begin a valid word. Then you can backtrack your permutation generator to the point where that tile was placed.

4.6 Assessment

Steven Gordon's generator is the generator of choice now. It uses more memory than Appel-Jacobson, but that is an insignificant downside nowadays. On the upside, it is about twice as fast. Cherry's algorithm is best when RAM is at an absolute premium but ROM is cheap. A handheld game machine could have such characteristics. The bit-parallel algorithm may be best on wide word machines, or if integer SIMD operations are available. Intersecting two bit vectors requires just 7 bitwise ANDs of 64-bits words. However, it is not suitable if incremental generation is needed.

MAVEN still uses the Appel-Jacobson algorithm. Historically, MAVEN ran on microcomputers that did not have 2.5 megabytes in total, so the GADDAG algorithm was not available. MAVEN should switch if it wants top speed.

4.7 Architecture

There is a significant implementation decision beyond the choice of the algorithm. Because Scrabble has several phases of play, you need to structure the move generator so that it can be used in every phase. MAVEN's architecture is helpful.

MAVEN's move generator generates and scores each move, then passes it to a caller-supplied handler function for further processing. This "handler function" architecture facilitates reuse of the move-generator module in a variety of search engines. All search engines in MAVEN use the same move generator with different handlers.

MAVEN's generator is not reentrant, but there is no need for that capability. If Scrabble required recursive search then reentrant move generation would be necessary. However,

since excellent plausible move generation using static evaluation is feasible in Scrabble, recursive generation is not beneficial.

Chapter 5 – Rack Evaluation

At first, MAVEN's advantage over human experts was attributable to the fact that MAVEN evaluated the tiles left on its rack after a move better than humans did. This chapter describes that capability, which remains largely unchanged from its original form of the year 1986.

The first section describes the problem. That there is a problem can be seen from the fact that top human masters average 33 points per move despite inaccurate move generation, whereas a perfect move generator with no rack evaluation will average only 30 points per move.

The next section describes the requirements of a rack evaluator. Some of the requirements arise out of the problem statement, and some arise out of software engineering goals that make the rack evaluator faster and more flexible.

The third section describes the architecture that MAVEN uses. The architecture leaves two points of flexibility for tuning purposes. First is the concept of a tile pattern, which is a set of tiles left on the rack. Second is the concept of a pattern value, which is a value that is accumulated into the evaluation when a tile pattern matches.

There follows a section that describes the actual tile patterns in the original 1986-era MAVEN, and then some extensions that increased MAVEN's robustness in extreme situations.

In many domains, there is an established theory that describes the values of evaluation patterns. However, Scrabble is not one of those domains. The tile patterns were largely self-evident, but there was no reliable theory to guide their settings. Accordingly, MAVEN used a tuning process that automatically set the values of parameters. This process is described in the fifth section.

The sixth section gives the so-called "Basic" evaluation model, which is a humanly executable model derived from MAVEN simulations. This model differs from MAVEN's internal representation, but agrees with the moves selected by MAVEN with almost perfect correlation.

Some situations that can be viewed as rack evaluation problems are specifically excluded from consideration in this chapter. The nature of these exclusions is described in the seventh section.

Next, we will cover the methods used to validate these parameters. Several people have approached rack evaluation from different directions, and we have all come to the same conclusion: that it is hard to surpass the Basic evaluation model.

Finally, we close the chapter with a historical assessment of the significance of MAVEN's rack evaluation methods.

5.1 Problem

It is clear that there is a need to evaluate rack leaves. A program having no bias against bad tiles will often hold racks like IIIUVVW, because it plays away its good tiles until only bad tiles remain. After all, there are *more* moves that use good tiles, so without any effort specifically directed at playing bad tiles it is certain that the program will play more good tiles.

Holding bad tiles is worse than it looks. You might figure that if you held IIIUVVW then you could take a few turns to play off the tiles and you would be free and clear. However, the situation is not so simple for several reasons:

1) You cannot afford to score badly for a few turns. For instance, MAVEN's average score with rack leave evaluation is 5 points per turn higher than its score without rack evaluation. Obviously, IIIUVVW is worse than an average random rack, so suppose that it costs 8 points per turn to play off these tiles. A few turns of that will leave you well off the pace. You can exchange if the bag holds at least 7 tiles, but exchanging costs a whole turn.

2) The assumption that you can replace these with random tiles from the bag is mistaken. For a simplified model, divide the tiles into two equally likely groups: the good tiles and the bad tiles. If you play two tiles off your rack, with the goal of clearing the drek, then you stand to draw one good and one bad tile. Next turn you face the choice: should you keep your good tile, or play it to help get rid of other bad tiles or to get a higher score? Odds are good that you must play it, meaning that over two turns you have made little progress towards balancing your rack.

3) Accordingly, it is likely that you will hold a significantly negative rack for more turns than you think. If you scored 5 turns at 8 points below average, then you have a net loss of 40 points over the sequence.

This example shows how bad racks act like *attractors* in the space of all possible racks. Once you hold IIIUVVW, you are guaranteed to have another bad rack next turn. The bad tiles on a rack devour the good tiles. Additionally, good tiles have a repelling quality. If you have an opening rack of ADEINOT, you would start anagramming gleefully, looking for the bingo that must be there. Alas, while you have several six-letter words, there are no sevens. Should you play a six-letter word? That is what a program that maximizes score alone would do: DETAIN (8D) scores 18 points, keeping O. You would be better off to exchange the O, keeping ADEINT. Exchanging scores zero, but retains a huge bingo potential.

5.2 Requirements

The overriding goal is to cause MAVEN to trade off the current score against future scoring. The balance between present and future scoring is at an optimal level when total scoring is maximized over the whole game. Therefore, we want to keep good tiles, but we do not want to give up too many points as a consequence. To achieve this goal it is necessary, at a minimum, for the evaluator to express all of the basic tradeoffs between tiles. This is the ability to discriminate.

Please note that the resulting evaluation function cannot be a first-order function of the tiles. Consider, for an extreme example, the rack EEEEEEE. Obviously, the first E cannot be too bad, but each successive E hurts, and probably hurts more than the one before. Another example is QU. Obviously a Q is a bad tile, since you usually need a U to play it away. In addition, a U is not a great tile. Nevertheless, the QU *combination* is about neutral.

One technical point is that the evaluation is expressed in terms of the tiles kept, rather than the tiles played away. These seem equivalent, but a subtle difference is hugely important to the ability of the program to learn tile values. The difference is that the value of a tile played away depends heavily on what was on the rack in the first place, whereas the value of a tile kept is largely independent. For example, if you hold EE and play away an E, then that is good, improving the rack by about 3.5 points. If you hold E and play away the E then that is bad, worsening the rack by 4 points. These perspectives are rationalized by observing that the value of keeping a single E is 4 points, whereas the value of keeping two E's is 0.5 points.

Finally, the rack evaluator should be sufficiently fast that it does not interfere with move generation speed. Speed might not seem like a necessary condition, since MAVEN is much faster than seems necessary. However, the need for speed arises from the CPU burden of simulation, as described in Chapter 10.

5.3 Architecture

5.3.1 Ability to Discriminate

Since the evaluator will definitely encounter racks consisting of individual tiles, it cannot achieve its discrimination goal without having values for keeping each tile.

The examples of QU and II show that combinations of tiles may have joint values. Therefore, the evaluation uses a list of *combinations* of tiles, each combination having an evaluation parameter. In MAVEN, a tile pattern may include from 1 to 7 tiles of any identity including duplicates.

Each tile pattern has an associated value, which is added into the evaluation of racks that match. Note that the addition is independent of other patterns that might match. Let us take an example. Suppose we want a QU to have value 0, and the value of a Q is –13 and the value of a U is –5. What value should the QU pattern have? The answer is such that Q + U + QU = 0, so QU is valued at 18. This value does not mean that racks that contain QU score 18 points above average, nor that the score of a game that contains the QU is 18 points above average. It *does* mean that holding a Q and U together is 18 points better than holding them in separate turns.

Note that the evaluator is linear, but combinations enable modeling of complicated factors [29]. The combinations may include any group of tiles, and the list may be arbitrarily long. An implementation trick, described in the next section, prevents the size of the pattern list from affecting performance. A later section lists the patterns actually used.

5.3.2 Performance Requirement

The requirement to express the evaluation in terms of tiles kept implies that we need to evaluate a group of from zero to seven tiles. Moreover, the tiles are the ones on the rack at the start of the move, so there are at most $2^7 = 128$ possible combinations of tiles that figure into the rack evaluations of moves on a single turn.

The average number of moves per turn is about 800, and since this exceeds 128 it transpires that we benefit from pre-computing the values of all possible rack leaves. Accordingly, in MAVEN the rack evaluations of all 128 possible rack leaves are pre-computed and stored in a table.

The index of that table is an integer between 0 and 127 that uniquely identifies the combination of tiles still on the rack. A bit mask is initialized to 127 and a bit is cleared for each tile played away. The only complexity is the treatment of duplicated tiles.

The pre-computation of rack values means that MAVEN's rack evaluation cost is largely independent of the number of patterns used in evaluating racks. The HITECH chess engine, which was state-of-the-art when MAVEN debuted, inspired this design [23].

Moreover, the total rack evaluation cost is minor, consisting only of a few bit mask operations and an array lookup for each move generated. Therefore, the speed requirement is met.

5.4 Necessary and Sufficient Tile Patterns

The first version of MAVEN's rack evaluator included a value for every tile (Blank, A, B, C, ...) and every duplicated tile (Blank-Blank, AA, BB, ...), and every triplicated tile (AAA, DDD, EEE, ...). For quadruples and higher the value used for triples was taken. There was a value for QU, since these are individually bad tiles, but not bad when held together.

This model alone was sufficient for championship caliber play when MAVEN debuted, owing to the fact that MAVEN's evaluation parameters were significantly more accurate than the heuristics employed by humans. Nevertheless, those patterns alone are logically unsatisfying, and some extensions proved beneficial.

Some value should be associated with drawing a tile from the bag. For instance, the rack leave AET should be thought of as "AET plus 4 tiles from the bag." This is important because AET may be compared with "AERT plus 3 tiles from the bag." MAVEN's tile turnover value is the average of the values of the unseen tiles. Thus, if the tiles in the bag are better than the tiles in the rack then MAVEN has a reason to play its tiles, but if the tiles in the bag are worse then MAVEN prefers to keep its tiles.

Another heuristic is called "Vowel / Consonant Balance". For example, NRST is not as good as the sum of its tiles because there are too many consonants. At first MAVEN included bonuses for all possible counts of vowels and consonants. This improved scoring by 3 points per game. Later, we switched to a dynamic bonus that depends on the ratio of vowels to consonants among the unseen tiles. This measure does not improve performance in any noticeable way, since it only really matters in the infrequent case where the distribution of unseen tiles is significantly out of normal. Nevertheless, MAVEN

retains it because it does occasionally result in better moves in the pre-endgame stage, and it does not hurt any other time.

One measure we tried was to include a term in the evaluation function for every pair of tiles (i.e., 351 pairs). We found that there was a positive value associated with almost every Vowel + Consonant combination, and a negative value associated with almost every Vowel + Vowel or Consonant + Consonant combination, whereupon the simpler Vowel / Consonant balance term seemed like a better choice.

The second heuristic, "U-With-Q-Unseen", is sometimes important. The idea is that holding a U when there is a Q unseen is better than it seems because a U-less Q costs 12 points, whereas a QU together is neutral. A linear function implements this concept.

The third heuristic is called "First-turn Openness". This heuristic expresses the theory that it pays to play the first turn "tightly" by using a few tiles only. The reason is that the reply to the first move can be awkward if the opponent has no access to double word squares and few letter to form bingos. We have computed a table of bonuses associated with starting games by playing 2, 3, ... up to 7 tiles. MAVEN has implemented this feature, but we are uneasy about it because it is in MAVEN's general interest to open the board against humans.

Finally, a recent refinement adjusts the values of tiles according to the probability of drawing a duplicate out of the bag. The theory is that if a tile is in plentiful supply then we should give it less weight. MAVEN has a linear function that implements this factor. Usually this factor has no effect, but it does give MAVEN a reasonable bias when a situation is extreme. This is called the *tile density* evaluation.

5.5 Parameter Tuning

What remained was to estimate all parameters. Various alternatives were considered, like simulated annealing [24], which was all the rage in those days (i.e., 1986). We decided to learn parameters through a feedback loop. The idea was to play games, then value each combination according to the impact that it had on future scores.

We want to choose a move that maximizes the sum of the score of the turn plus the value of the rack. With the goal of trying to maximize scores over the game as a whole, we can define the value of a pattern to be the amount by which your score increases over the duration of the game by keeping that pattern on your rack after a turn. We can create a feedback system that directly measures the value of a pattern.

For example, suppose that the rack leave of a particular move is EQU. This rack contains the following tile patterns: E, Q, QU, and U. Therefore, this turn is an observation for four parameters. The future score is the difference in score between the side to move and the opponent over the rest of the game. After collecting many such observations, you have an over-constrained system of linear equations. You can solve that system in a number of ways. The solution we used at the time exploited domain-specific properties. Possibly, we should choose least squares, following the recommendation by Michael Buro [29].

Let us take an E as the pattern in question. The feedback system keeps a record of the results from every situation in which an E remains in the rack. We define an observation as the difference between the player's scores and the opponent's scores for the period of time over which an E remains in the rack. For example, suppose we observe the sequence of plays from Table 5-1.

Rack	Tiles Played	Tiles Left	Score	Opponent's Score
ABCDEFG	ABDC	EFG	20	17
EFGWXYZ	FXY	EGWZ	35	43
EGWZRST	WZ	EGRST	30	29
EGRSTRE	REGRETS	-	72	

Table 5-1 Example of Rack Parameter Tuning

In this case, the E that remains after the first turn is on the rack for 3 more turns. The value of this observation of the E is the difference in scores over the next 3 turns for each side. The turns start with the opponent's next score, so the value observed is -17 + 35 - 43 + 30 - 29 + 72, plus the values of any tiles left on the rack at the end, but here there are none.

It is important to regard an observation as spanning multiple turns because some tiles can persist on the rack for several turns. A Q, in particular, often sits for several turns while you await a U or QAID, QAT, FAQIR, or another rare word that plays the Q without a U. During that time you are essentially playing with 6 tiles against an opponent playing with 7, so you have a significant disadvantage on every turn.

Another situation that can last for a while is the possession of duplicated vowels. Let us say that you keep AA on your rack. You have a significant chance of drawing another A to make triplicates. If you play away one of the A's next turn then you may be back to duplicates. The I is an easy tile to play, but it is hard to play two I's at once, so II is a terrible rack leave that tends to accumulate additional bad tiles. It is also hard to play two U's at once, so U's tend to accumulate, and keeping UU is even worse than keeping II because playing a U is harder than playing an I.

One final situation. Humans had historically given the blank a value of 40 points, meaning that you play the blank if you have a move using the blank that scores 40 points more than the highest play not using the blank. The basic point is that you should hold blanks for making bingos, which typically score 75 or more points, where a typical play is 35 or fewer points. As it turns out, blanks are worth closer to 25 points. The reason is that the high value of a blank often prevents it from being used for several turns as you await bingo tiles. During those moves, you must effectively play with only 6 tiles, which hurts your scores. A lower value for a blank allows the blank to be used for crucial non-bingo moves, such as dumping a Q, or using a Z to hit a triple word square.

One key point is that patterns match independently. For example, the value of a rack containing EE equals the value of EE plus the value of E. Therefore, if you observe that holding EE results in a 1 point improvement in scoring, and holding E results in a 4 point improvement, then the value of E is 4, and the value of EE is -3. In general, to compute the value of a pattern you must subtract the value of all other patterns that necessarily match.

Each game contains many observations of many patterns, so if you automatically play games overnight then sufficient data accumulates for firm conclusions. Computers nowadays are fast enough that MAVEN can tune itself to changes in the dictionary in less than an hour.

Note that this is not an online learning system. In other words, the values do not change the program's play while data is collected. It follows that the data collection task is stationary. After playing overnight, MAVEN assigned each pattern the average value of all observations. This changed the play of the program, and that changed the observed values in a new overnight run. Then MAVEN repeated the process, which converged on the third iteration.

Instead, we could have used temporal differences (TD). MAVEN's evaluation model is W * P, where W is a vector of weights, P is a Boolean vector that indicates which patterns match, and * is the dot product operation. Then play games, using temporal differences to adjust the weights between every turn. That is, backpropagate a fraction of the evaluation of the future of the game into the evaluation of the present. Would that be simpler?

It would be simpler, but it did not work when it was tried in MAVEN. MAVEN tested several feedback systems but the weights never came out right. The weights seemed compressed towards zero compared with the true weights. For example, MAVEN's weight for an E is about 4.2, whereas TD learned a value near 2.0. MAVEN's weight is indisputably better, which MAVEN proved in a self-play test. Why did TD not work?

Our explanation is our deficient exploration policy: an on-line learner learns to avoid terrible racks like III before their values have converged to their true (and truly awful) values. It is not that learning stops, but the program can almost always choose not to keep III, if only by exchanging, and therefore the learning system is deprived of examples that would move the weight towards the true value.

A comparison between backgammon and Scrabble is in order, since TD works like a charm in backgammon [38], whereas it did not work in Scrabble (which has a *much* simpler parameter space). Sutton's theoretical arguments show that TD should work if given occasional opportunities to "explore" paths that seem sub-optimal [30]. In Scrabble you are almost never *forced* to keep III, so learning can stop even the start of an awful convergence. In backgammon, by contrast, the dice often force you to accept all sorts of positional defects, even if you play ideally. The lesson is that in applying TD you must ensure adequate exploration.

5.6 Basic Rack Evaluation Model

MAVEN's actual tile valuations have six decimal digits of accuracy, so they do not form a theory that is useable by a human. However, one does not need such precision. If MAVEN's rack evaluation parameters were rounded to the nearest 0.5 then the quality of play would be indistinguishable. The observation that this level of precision is sufficient is due to James Cherry, who employs such truncated weights in ACBOT [25].

Table 5-2 shows tile parameters computed by Nick Ballard (a Scrabble master, but better known as a backgammon World Champion) and Charlie Carroll. They computed this

table by designing pairs of MAVEN simulations that isolated the value of a single tile. This is tricky because adding a single tile to a rack changes the tiles *and* the vowel-consonant balance. Ballard and Carroll computed the values by carefully crosschecking simulation results.

Tile	First Tile	Second
A	+0.5	-8
B	-3.5	-8
C	-0.5	-7
D	-1.0	-6
E	+4.0	-3.5
F	-3.0	-6
G	-3.5	-10
H	+0.5	-6
I	-1.5	-10

Tile	First Tile	Second
J	-2.5	
K	-1.5	
L	-1.5	-6
M	-0.5	-6
N	0.0	-5.5
O	-2.5	-8
P	-1.5	-6
Q	-11.5	
R	+1.0	-9

Tile	First Tile	Second
S	+7.5	+1
T	-1.0	-6
U	-4.5	-12
V	-6.5	-8
W	-4.0	-8
X	+3.5	
Y	-2.5	-10
Z	+3.0	
?	24.5	+12

Table 5-2 "Basic-1" Rack Evaluation Model

Table 5-2 was published in *Medleys* [17], and dubbed the "Basic" evaluation model. They are here because they illustrate the actual state of MAVEN's rack evaluator in the year 1987, and to show normal rack evaluation priorities. In this thesis, we will use exactly these rack evaluation parameters for expository purposes, bearing in mind that MAVEN's implementation is slightly different.

The "First Tile" column shows the bonus added to the rack evaluation when a tile is present in the rack leave. The "Second" column shows the bonus added if there are at least two. Note that the second bonus is *in addition* to the first.

Table 5-3 gives Ballard and Carroll's vowel-consonant balance heuristic. The data show that having a roughly equal number of vowels and consonants is good, and one should give a slight preference to consonants. As described above, the current MAVEN does not use this type of function, but it did around 1988. This table of parameters has the advantage of being humanly executable, whereas MAVEN's current function has the advantage of improving quality when the vowel-consonant balance of the bag is extreme.

		CONSONANTS						
		0	1	2	3	4	5	6
V	0	0	0.5	1.5	0	-3.5	-6	-9
O	1	-0.5	1.5	1	0.5	-2.5	-5.5	
W	2	-2	-0.5	0.5	0	-2		
E	3	-3	-2	-0.5	1.5			
L	4	-5	-4.5	-3				
S	5	-7.5	-7					
	6	-12.5						

Table 5-3 Basic Vowel-Consonant Balance Table

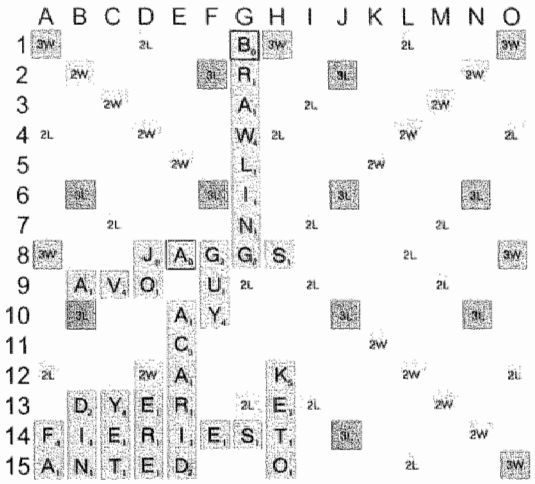
Table 5-4 shows a few values for compound tile patterns. Note that these values are in addition to the values of the individual tiles. MAVEN has many such patterns, but we show only a few. The values of compound patterns can be hard to interpret in isolation, because of interactions with patterns that share tiles. The patterns in the table are comparatively easy to understand; they represent a group of tiles that often occur as a substring of a word.

Pattern	Value
QU	+17.0
GIN	+6.0
IVE	+1.7
OTU	+3.0

Table 5-4 Sample Compound Tile Patterns

5.7 Specific Exclusions

Some concepts that arise in Scrabble are not encoded in MAVEN's rack evaluator. These have to do with how the values of tiles are interrelated to specific situations on the board. This thesis contains a number of examples, but one more will not hurt. Position 5-1 was published by Nick Ballard in *Medleys* [27].



Position 5-1 Surprising Rack Synergy

Table 5-5 shows the moves considered by the consensus panel. ZEBU (1E, 36, AHIN) is the conventional move in this position, since higher scoring moves either keep the U or leave the triple-word line open for only a few points more. KHAZEN is exceptional: it scores 8 points more than ZEBU, but the IU rack leave has such bad synergy that 12 points is probably required for adequate compensation. One move having a nice hidden advantage is BIZE (1G, 36, ANHU), which is as good as ZEBU despite keeping a U instead of an I because it sets up our H at 2J.

Word	Spot	Score	Rack
KHAZEN	12H	44	IU
HAZIER	2B	40	NU
BAIZE	1G	39	HNU
ZEBU	1E	36	AHIN
BIZE	1G	36	AHU
ZANIER	2B	34	UH
BIZ	1G	33	AEHU
NAZI	B8	33	AEHU
NAZI	6D	33	EHIU
UH	10A	26	AEINZ

Table 5-5 Good Moves for Position 5-1

Not surprisingly, ZEBU was the consensus choice when this position was given in *Medleys*. Probably most readers wondered what surprise Ballard had in store for them. The surprise is that UH (10A ,26, AEINZ) is the best play. The advantage of UH over ZEBU is only 3 points, but the advantage looms large at this score because UH's rack leave AEINZ has a surprisingly high bingo chance: about 15%, which is much more than any other move. This is because the AEINZ rack leave makes many bingos ending in IZE through the letters in BRAWLING. Moreover, when you do not bingo, you still have a big play for the Z. Since the B at 1G is a blank, the opponent leaves the top row open 74% of the time, whereupon BAIZE (1G, 39) can follow if nothing better turns up.

Synergy between the rack and the board occurs regularly, and it is tempting to try to cram such concepts into the rack evaluator. For example, add an EIZ tile pattern and away you go! However, the EIZ tile pattern (which MAVEN has) does not increase UH to the level that is appropriate in *this* position. The EIZ tile value represents the average value of that tile pattern, which is too low.

To properly understand how tiles interact with the board requires simulation, so we will postpone discussion of synergy until Chapter 10.

5.8 Testing and Validation

This section will evaluate the rack evaluator on two criteria. The first criterion is whether the set of patterns that it encodes is sufficient. The second criterion is whether the patterns are valued appropriately.

5.8.1 Sufficiency of the Tile Pattern Set

The first criterion is whether the pattern set is sufficient. Well, we can answer "Yes" because MAVEN plays a championship-caliber game, but that answer is somewhat superficial. We do not mind judging by results in this situation, but the deeper question of why the pattern set is sufficient deserves an answer, too.

One key property of our tile set is that every rack receives a distinct value, and that value depends on the context (specifically, it depends on the unseen tiles). This property is required of any evaluator that is sufficient for championship play. While it is nice to see that the rack evaluator is capable of discrimination, that property alone does not guarantee that the evaluator is sufficient. MAVEN's evaluator includes several components that possess this property on their own, yet I am not considering removing the other pieces!

Another consideration is that MAVEN implements all of the factors that human experts use. The writings of human analysts mention four concepts: the quality of individual tiles, vowel/consonant balance, the desirability of splitting up duplicates, and synergies

between specific pairs of tiles. All of these concepts are in the rack evaluator, so it seems likely that MAVEN incorporates the necessary concepts.[20]

Still, one has to wonder how we know that humans actually have all of the necessary concepts. After all, the computerization of Scrabble transformed humanity's theory of the game, so why should they not be in error on this topic?

Ultimately the answer is that the author tried for a whole year to improve the rack evaluator, and was unable to do so. During that year, MAVEN tested many tile patterns, and only found a few patterns that were worth having. For example, the patterns ING, OUT, IVE and IZE were beneficial. In addition, the vowel pair IU is sufficiently destructive that inclusion is beneficial, but other than a few such patterns, nothing helps much.

One of the reasons why longer tile patterns make little difference is that they are unlikely to arise. The value of the ING pattern is large, but MAVEN will only rarely hold all three tiles at once. Moreover, in few of these will it pay to keep those tiles, since keeping all of them means the move must be made out of the remaining 4 tiles. Even then, the best move may be clear without reference to the pattern value of ING.

Now you can see why MAVEN's tile patterns are sufficient. They included everything of high frequency, and every short pattern. Longer patterns arise too rarely to make a difference.

5.8.2 Optimality of Pattern Values

We are convinced that MAVEN's parameters are close to optimal. Several tests confirm this.

5.8.2.1 Self-Play Testing

We conducted tests pitting MAVEN against MAVEN with slightly different rack parameters. The number of games needed to obtain statistical significance with this method can be large if the parameter sets are close together, but gross differences are readily detected. For example, if you tested a value of 2.0 for the E then you would quickly find that performance has deteriorated. I would not assert that the tests were exhaustive, since there were limits to my patience, but we have done the testing for significant tiles such as E, Blank, S, and Q.

One finding from that test is that there is a broad, flat region atop the parameter space. That is, changing the value of individual patterns by a point or so makes no difference to playing strength. You could not change the value of a pattern by 3 points and get away with it, but small changes are insignificant. There are two reasons for this.

First is that the two highest-scoring moves in a position usually differ by more than a point. If so, then a one-point change in the evaluation cannot change anything. Even when the top two moves are close, you still need two specific conditions to apply. You need the pattern to match for one move but not for the other, and you need the evaluation

[20] Of course, these concepts may have a richer form in human understanding than in MAVEN's implementation.

difference to be supportive of the inferior play. The entire set of conditions is unlikely to apply simultaneously.

Second, suppose that you do make an error and choose a move that is one point inferior. What of it? One point changes the outcome of a game less than 1% of the time. It is hard to measure such differences.

This reasoning shows why MAVEN's rack evaluator leads to championship caliber play, at least with respect to rack evaluation. It is because the differences between moves tend to be larger than the standard error in the measurement of tile values. Thus, the ranking of moves is usually right, and when an error occurs it usually reduces winning chances by little. (Again, the caveat is that there are ways to make errors besides simply misevaluating the quality of a rack.)

5.8.2.2 Word Counts

The conclusion of the last section is disturbing. If we can vary the tile values without changing the outcome, then why would we believe that the parameters are near optimal?

Qualitative criteria show that the relative order of the tiles is correct. These criteria involve counting words and making inferences about how that would affect tile valuation. For example, consider the vowels. Table 5-6 shows how many bingos use each vowel. Is it surprising that E is more valuable than the other vowels?

Vowel	Bingos
E	34249
I	26089
A	24855
O	19047
U	12855

Table 5-6 Bingo Counts by Vowel

Another issue that can help to rank tiles is the relative frequency. For example, the II is more damaging than AA, and yet A and I have the same frequency in the tile pool. Is it any wonder that A is more valuable than I?

Comparisons of word counts establish a collection of relative rankings. Although these rankings are insufficient to totally order the tiles, they do validate the Basic model.

5.8.2.3 Hill-Climbing Search

Another approach is to do a hill-climbing search of the parameter space. In 1994, Steven Gordon published the results of such a search [28]. Gordon's method was to modify parameters by steps of 0.5 using a competitive co-evolution strategy. The results agree with MAVEN's parameters (on the whole) to within 0.5.

5.8.2.4 Other Developers

Other Scrabble developers (e.g., Jim Homan on CROSSWISE and James Cherry on ACBOT) have reported similar results. For a full understanding of the method and the reported results, we refer to the references [25, 48].

5.8.2.5 Simulations

Perhaps the most convincing evidence of optimality is the Basic model. Carroll and Ballard constructed the Basic model by external observation of MAVEN's behavior. Yet their rack parameters agree with MAVEN's internal values. This shows that MAVEN's rack evaluator is internally consistent. Upon learning the details of simulation in Chapter 10,

you might think that there is an element of self-fulfilling hypothesis in validating MAVEN's rack evaluator by using MAVEN simulations. After all, MAVEN chooses moves during simulations using MAVEN's rack evaluator, and MAVEN evaluates the endpoints using the same rack evaluator again. Shouldn't MAVEN simulations confirm MAVEN's rack evaluator?

Well, it would confirm the rack evaluator's values if the rack evaluator were *externally* consistent, which means "consistent with the Scrabble space." The dominating factor influencing the outcome of a simulation is not the tiles kept at the endpoints, but the scores achieved along the variations, so to have an evaluation function consistent with such a process is significant.

5.8.3 Overall Assessment

MAVEN's rack evaluator is thorough and accurate. It is genuinely hard to surpass without simulating how racks interact with the board position.

5.9 Historical Significance

Before MAVEN's results, humans used rules of thumb that were significantly in error. The rules of thumb failed the test of sufficiency, in that they were unable to discriminate between all possible racks. For example, Wapnick recommended [8] sacrificing two to three points for each tile played, subject to his other recommendation to play the S only if you score 8 points more and the blank only if you score 40 points more, and take 13 points less if you can play the Q. This advice essentially values every tile except Q, S, and blank at –2.5 points. Which is about right if the tile is a J, but substantially wrong if the tile is E.

Another human goal was to split up duplicates. Wapnick recommended giving up 5 points to split up a duplicate, except SS and Blank-Blank, of course. While you should be willing to forego more than 5 points for any duplicate except EE, this recommendation will result in good play; five points will overturn most move decisions.

This is an instance of an important principle, which I will dub the "Flatland principle:" evaluation parameters need not lie exactly on their true optima, but merely need to lie in a region nearby such that the gradient of skill is essentially zero as that parameter changes. The author's experience suggests that evaluation functions often have "flatlands" near their optima, and this is certainly true in Scrabble. The reason is that a small change in an evaluation parameter will change only a small fraction of moves, and the number of points lost is necessarily small. In effect, as an evaluation parameter nears its optimum the gradient of skill is like a quadratic function, since errors decrease in both frequency and size.

Vowel-consonant balance was another goal. Wapnick recommended docking 2 points for every tile deviation from ideal, which he regarded to be a 0-0 vowel-consonant balance. The gradient of this function is a little too steep near the ideal, and too shallow farther away, but it improves over having no function at all. Alas, it has the nasty side effect of further reducing the value of an E.

Humans also had a theory about awkward pairs of consonants. The theory went that specific pairs of heavy consonants played poorly together, so there was value to splitting

them up. MAVEN was unable to detect the effect of this refinement, possibly because pairs such as PZ occur rarely, and when they do it is highly likely that one of the two will be played anyway. Also, vowel-consonant balance heuristics subsume many such patterns. In contrast to the consonant pairs, however, is a pair of vowels: MAVEN proved a significant negative value for the IU pair. Other programs, however, do appear to incorporate pairwise synergy values [31].

5.9.1 Classical Human Theory in Action

For example, here is an assessment from Joe Edley: Lisa Odom played KHI (18, EIOPT) in a game against Edley. Edley thought KHI was weak, and recommended KEPI (12, HIOT), with POTICHE (14, I) as second choice. Table 5-7 shows an evaluation of these moves under classic turnover theory. Note the strong bias towards long moves.

Word	Score	Leave	Turnover Theory	V/C value	Total
KHI	18	EIOPT	2 * 2.5 = 5	-2	21.0
KEPI	12	HIOT	3 * 2.5 = 7.5	0	19.5
POTICHE	14	I	6 * 2.5 = 15	-2	27.0

Table 5-7 Turnover Theory Illustrated. Do Not Do This!!

Nowadays, Edley would not make such a recommendation, so greatly has positional understanding grown since 1991. This recommendation is so far off the mark that you do not even need to see the board. However, back then, concepts such as turnover still held a powerful grip on players, and notions like the Basic model were just beginning to gain credibility.

Based on score and rack leave, KHI is vastly superior to any other play. Compare KHI with KEPI and POTICHE using the Basic model, as shown in Table 5-8.

Word	Score	Leave	Basic-1 Theory	V/C value	Total
KHI	18	EIOPT	4 - 1.5 - 2.5 - 1.5 - 1 = -2.5	+1	16.5
KEPI	12	HIOT	0.5 - 1.5 - 2.5 - 1 = -4.5	+2	9.5
POTICHE	14	I	-1.5	-1	11.5

Table 5-8 Same Example Using Basic Model

Lest you think it is self-destructive to choose moves without considering the situation on the board, please be reassured that it is less destructive than playing in accordance with the turnover theory. (Actually, as the next chapter will show, choosing moves without considering the board is essentially what MAVEN does, and that procedure is likely to work.) Anyway, to satisfy skeptical readers, simulations showed that KHI was even more superior than the table shows. KHI was 11 points better than KEPI, and 20 points better than POTICHE. In the actual position, POTICHE had the drawback of opening a triple word square.

5.9.2 But Humans Have Changed

Nowadays, MAVEN's rack valuations are well known and widely accepted. The publication of the Basic model started the process off, and tournament successes by some of MAVEN's early users built interest. The author unabashedly takes credit for spreading

the gospel of rack evaluation, despite the fact that other computer programs (CROSSWISE, in particular) evolved similar rack evaluation theories at the same time. The difference between MAVEN and CROSSWISE was that MAVEN made its rack theory accessible to users, whereas CROSSWISE hid its theory.

For instance, if you asked MAVEN to list the best moves in a position, MAVEN would list the moves in order of evaluation, showing both score and rack leave. MAVEN could give you an English-language explanation of the difference between two plays, using the Basic model for expository purposes. (A fascinating feature, but one that is beyond the scope of this thesis.) These features, and the constant preaching by Edley, Ballard, Felt, Carroll, and other masters who believed in the system, made an impact. By contrast, if you asked CROSSWISE for the best plays in a position, it would show the list in order of highest *score*, which purposely hid what users really wanted to know.

Chapter 6 – Positional Evaluation

The last chapter gave an example of a move in which MAVEN recommends KHI over KEPI and POTICHE *without even seeing the board.* Surely, that is a risky practice! The shape of the board must have *some* influence over the quality of moves. In this chapter, we consider that influence.

The first section describes the approach MAVEN used in the late 1980's, and the surprising successes that MAVEN had with a simple strategy. The next section describes some theories that MAVEN tested, which were in common use by human experts of the early days. The third section outlines the final form of MAVEN's positional evaluator, and justifies the choice using qualitative reasoning based on the domain analysis. The fourth section describes MAVEN's triple-word square evaluator. We find that MAVEN has changed how humans evaluate positions, and the fifth section gives a brief outline of those changes. The chapter closes with three sections that describe evaluation functions for special circumstances.

6.1 Board Evaluation in the Early MAVEN

It is obvious that each move changes the board. At first, it was not clear what to make of this, so we decided to assign the board a value of 0. In other words, MAVEN assumed that changes on the board were neutral. We will revisit that assumption in a later section.

The only exception to the general rule that MAVEN's positional evaluations were zero concerned triple word squares. The earliest MAVEN included a penalty of 3 points for leaving a triple word square open (i.e., directly accessible without requiring possession of a hook tile).

The resulting program is heavily biased towards maximizing score + rack, and therefore the early MAVEN played a wide open, high scoring game. Humans criticized MAVEN for being too aggressive about opening the board, but it was hard for the author to see how this tendency hurt MAVEN. It seemed that MAVEN's opponents were hurting! Still, MAVEN's first tournament produced a long list of theories to test.

6.2 Human Positional Theory

MAVEN's first tournament proved that humanity was more vigorous about evaluating boards than MAVEN was. It is obvious to human experts that the board had a huge effect on scoring, and therefore controlling the board should have a beneficial effect on winning.

Human expert contacts described several positional considerations, which the next 5 subsections cover. Each subsection describes one positional theory, and the impact of that theory on MAVEN. This section concludes with some sweeping generalizations about human expert positional reasoning.

6.2.1 Opening the Board

One widespread expert theory is that "opening the board" (i.e., creating spots where bingos could be played) is bad, on the theory that the opponent obtains a first crack at

playing a bingo, and if the opponent has no bingo then he could "shut down the board." This seems like a promising "grand strategy," but it does not work in practice. We tried several functions that included significant understanding of bingo chances. All were worse than including no openness function at all. There is no general penalty for board openness. To understand board openness requires simulation.

Even if openness were slightly negative, it would be bad tactics to include such a factor in MAVEN, since the program never misses a bingo, whereas human masters miss 10 to 20 percent of bingos. Obviously, this makes up for any "first mover" advantage.

6.2.2 Vowel-Bonus Adjacency

Experts expounded the general principle of not placing vowels adjacent to bonus squares. The idea is that a high-scoring tile can be "double-crossed" on the bonus square, leading to a good score for the opponent. For instance, from Ozag and Lawrence [32] we learn that it is appropriate to sacrifice 6 points to avoid placing a vowel next to the double letter squares at 7G and 7I on the initial turn. This penalty turns out to be laughably large. When we used computer analysis to estimate this factor, it transpired that the largest such penalty is only 0.7 points! Most vowel-bonus square combinations had penalties of 0.01 points, meaning that you should avoid it only if there were no other consideration.

The worst case is placing a U to the right or below a triple-letter square. Such a placement has two drawbacks. First, if the opponent holds X then he is set up for a possible 50-point XU play. Second, if the opponent holds Q then he can rid himself of that unplayable millstone with a play scoring 31+ points. Playing off the Q in that fashion is a huge gain, since you get an above-average score using terrible tiles. Nevertheless, because the opponent is unlikely to hold the key tiles, only 7/10 of a point rides on the outcome.

MAVEN includes this factor in its evaluation function, but it serves as a tiebreaker. For example, on the first turn you can play BARED, BREAD or BEARD, all at 8D and all scoring 22 point and keeping the same rack leave. The best of the three is BEARD. BARED and BREAD leave a vowel next to double-letter squares at 7G and 9G, and BEARD does not. It is not worth sacrificing points, but it makes MAVEN look more polished when it has a choice.

6.2.3 Building a QAID

Before the "discovery" of the word QAT in 1993, the simplest way to get rid of a U-less Q was to play QAID. There are other words (FAQIR, QINTAR, QOPH among them) but they require either more letters or less frequent tiles.

Human experts explained the strategy of creating a QAID. The idea is that the Q and D are the only scarce commodities, so if you happen to have a U-less Q with a D then you should keep the pair together until you accumulate an AI from either the bag or board. This strategy shows how rack evaluation can implement complex positional theories. MAVEN tested the patterns QA, QI, QD, QAI, QAD, QID, and QAID in the rack evaluator.

MAVEN discovered that the whole theory is wrong. Instead of having a positive weight, the QD had a negative weight, which means that MAVEN had determined that it pays to split up the QD! The next few paragraphs explain why MAVEN reached this conclusion.

First, simply accumulating the tiles for QAID does not guarantee that QAID will be playable. You might sacrifice points for nothing.

Second, even if QAID is playable, you might not get a good score. QAID might only score 14 points. Consider that being stuck with the Q carries a penalty of 20 points, and the average score of a turn is 35. If you score only 14 then you have paid full price for the Q, despite the success of the QAID-building strategy.

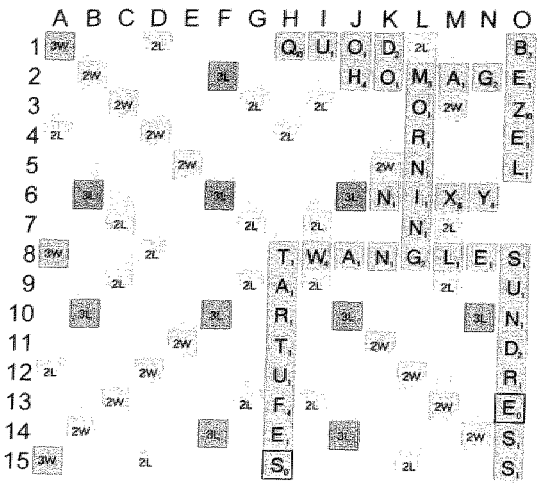
Third, the odds of success are lower than they appear. Let us split up the space of U-less Q situations in two. One piece will contain situations where all the U's have been drawn, and other will have situations where U's are still available. If a U is available, then the overriding priority is to turn over tiles to draw a U from the bag. Accordingly, hanging onto a D is a hindrance in that the D should be turned over to make room for a U. Besides that consideration, there is the likelihood that it will be easier to draw one U than to draw both an A and an I. Moreover, it will be easier to play a QU word than to play QAID, which might not fit anywhere. If no U is available then there must be 4 U's on the board. Statistically, that implies that the game is 80% completed. It is unlikely that you will be able to complete a QAID-building strategy before the game ends.

Finally, there is the general observation that Q has negative reaction with every consonant.

Thus, the strategy did not work, or at least this implementation did not work. The author would not have faith in any implementation that did not calculate whether QAID has a profitable place to play, or did not consider the tradeoff against reduction in the chance for QU plays.

6.2.4 Setups

Humans who hold several heavy tiles often find plays that create spots for follow-up plays. Such a move is called a setup. Position 6-1 is an example from game 9 of MAVEN's 1998 match against Adam Logan.[21]



MAVEN: I, J, N, O, O, P, R 194
Logan's last: QUOD (1H, 50) 207

Position 6-1 Example of a Setup

MAVEN's move was the beautiful double-double PORTION (11E, 36, J). PORTION keeps a J that can go on the triple-letter squares at 10F or 10J, making the crossword JO. For example, 10F JAR scores 53, as does 10H RAJ. We speak of PORTION "setting up" the J.

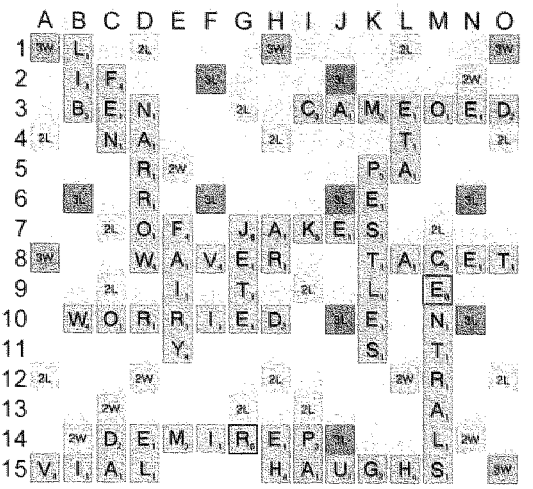
PORTION would be the best play even if it were not a setup. Thus, to some extent this example is misleading because it does not demonstrate how consideration of setups can improve on the Basic model. Nevertheless, the reader should see the potential.

MAVEN tried to find setups. The idea is to look for moves off of the tiles just played using tiles kept on the rack. If the potential score is high, then perhaps the setup is worthwhile. For example, if a setup promised 25 points then it would be worth little. But a 50-point setup is worth something.

However, there are still contingencies. Consider PORTION, keeping J. The biggest actual move using the J is 25 points. In order to get 53 you need to draw a vowel from the bag, plus possibly a cooperative consonant. Now this is not a stretch here, but what should MAVEN think about a situation where a setup can score 102 points if the draw includes an I? Consider the following example, from Wapnick's book [8].

[21] The entire game is annotated in Appendix B.

In Position 6-2, Stephen Fisher[22] eschewed several high-scoring moves to play TOD (O1, 12, BQSUZ). The reason? He keeps BZQU, so all he needs is to fish an I for BEZIQUE (N2, 102)! With 2 draws at 4 I's out of 19 tiles, Mr. Fisher has a good chance (39%). The author recalls that simulations do not quite justify this play on the grounds of point differential, but since Fisher is 100 points down this is just what he needs to get back in the game.



Position 6-2 Fisher's Fish for a Non-Bingo

In the end, the whole subject of contingencies derailed the effort. Consider the typical logic: if we draw a specific set of tiles, and the opponent does not block the spot, and we have no better play, then we have a specific score, which increases our equity by something, but it is unclear how much. I doubt that a robust evaluation function can be fashioned out of this type of reasoning.

The author's sense is that humans overrate such considerations. Nearly every board promises a large score if ideal tiles are drawn.

6.2.5 Fishing for Bingos

On an opening rack of AEIORST, you have no bingos. You can play the six-letter word SATIRE, but you should keep the S, if possible, so RATIO, keeping ES, comes to mind. That is what MAVEN would play.

A better move is probably to exchange the O, since the only tiles that do not make a bingo with the letters in SATIRE are JKOQYZ, or only 13 out of the 93 in the bag. Of course, the opponent will make his play, and you might not get a seven-letter bingo down, but there are tons of eight-letter bingos, and you might be a favorite to bingo even if you draw badly.

Such a play is called a fish. The example given above is about as pure a fish as there is, but there is a continuum of fishing plays that range from plays that are almost normal moves to outright speculative draws for a bingo.

MAVEN's inability to fish to draw a bingo is a genuine weakness. Humans fish quite often. Among weaker humans, the tactic is helpful even though many players overdo fishing. The SATIRE bingos are well known to weak tournament players, so fishing is one way to ensure that they will not miss a bingo.

[22] What a great name for a Scrabble player!

Curiously, strong players probably do not fish often enough. They distrust fishing because they usually play weaker opponents and therefore do not need to play the maximum number of bingos to win, provided that they do not sacrifice points on a regular basis. Consistently laying down good scores has merit against weaker opponents. However, if the anti-fishing bias carries over into games against peers, then that is probably wrong.

Fishing is complex to implement if you aspire to ideal play. It is easy to overdo fishing. For example, the play RATIO keeping ES is a good move, thanks to the ES retained and 12 points scored. The fishing play is only justified because the board is open for bingos, and the opponent is unable to block.

It is one thing to know that fishing can be beneficial, and another to generate all of the beneficial fishes and only those. Or, for that matter, even to improve on the Basic evaluator. Many moves have low turnover and keep good tiles, but only a small fraction of these are better than the best normal move. Even though we know the key metric—high bingo chance—it is still difficult.

One of the difficulties is to trade off fishing chances against normal scoring. The danger of evaluating using incompatible evaluation functions was first described by Berliner [34]. If the program orders the moves based upon score and rack on the one hand, and orders the moves based upon bingo chances on the other hand, how does the program produce an ordering that rationalizes the two lists?

Another issue is that many fishes are not selected based on point differential, but rather based on winning chances. A winning percentage evaluator could provide the basis for rationalizing two move lists. On the downside, this introduces a third metric into the mix.

Fishing is a challenging research topic. Top humans have largely ignored the issue by adopting the philosophy of rarely fishing. It may be possible to do a better job than MAVEN does. Chapter 10 will introduce a solution to this problem and more.

6.2.6 Playing Phonies

Top human players gain by playing phonies against their weaker human opponents, whereas MAVEN does not. The author recognizes that this is an issue. MAVEN is designed to play top champions directly, whereas tournaments are decided by play against the field, which will be composed of players much weaker than MAVEN is. Historically, a player having a rating around 2050 (pre-tournament) usually wins the National Scrabble Championship (NSC), but the average entrant is only a little over 1900 strength.

How significant is this factor? Human masters play few phonies. First, they are usually winning against their weaker opponents, so phonies are not even required. Second, when they are losing it will often not be a phony that gives them their best chance. Third, any phony they do play might be challenged. Finally, they might get away with a phony and lose anyway. What is the ability to play phonies worth? It is unclear, but the gain is limited because when a player overuses phonies then his opponents challenge more often.

The best possible design would endow MAVEN with the ability to play plausible phonies. However, do not count on the program being as effective as humans are at judging whether an opponent will challenge.

MAVEN does not play phonies, but there is a curious compensating factor: human opponents hesitate to play any word that they are not 100% sure of. In one tournament game, a player pondered playing a bingo for so long that he went 4 minutes overtime. The bingo was good, but he lost the game on the 40-point penalty for excess clock time.

Against other humans, the attitude is different. Some players (myself included) will slap down a plausible word that they are not sure of, thinking, "Now it's the opponent's problem."

6.2.7 Creating Bingo Lines

When MAVEN trails, it is often best to create new lines for bingos. MAVEN does not do so deliberately, so it can fail to create chances to come from behind. Of course, MAVEN's normal style is wide open, so whether this is a significant shortcoming is open to doubt. MAVEN has come from behind in many tournament games.

6.2.8 Defensive Play

When one side is so far ahead that his only fear is that the opponent will score a bingo, then "shutting down the board" is an important concept. Humans criticize MAVEN for not implementing this concept.

It is not that the author has not tried, but the question of how to hold onto a lead is harder than human positional theory allows. It is hard to decide whether to sacrifice point differential for defensive considerations. The typical human instinct of trying to block every bingo opening as soon as a 50-point lead is achieved is overly simplistic. When playing against top-level experts (human or computer) such a strategy is likely to backfire.

MAVEN is not as defenseless as some players think. MAVEN's favorite technique is to continue to lay down big scores, so that the opponent will lose even if he does bingo. If you implement only one policy then this is definitely the right choice because it is the most widely applicable policy. It works in the early game, middle game, and pre-endgame.

There is a side point concerning the decision to implement only one policy. Many people have advised MAVEN to implement one policy if it is 50 points down and a different policy if it is 50 points up. The author's experience suggests that this is unlikely to work well. The stumbling block is the "blemish" near the dividing line between the two policies. Berliner described the issues when creating his Smooth, Non-linear Application Coefficients (SNAC) architecture for evaluating backgammon positions [34]. SNAC allows an evaluation function to implement subdomain experts, but requires the evaluation function to integrate them by using continuously changing state variables called *application coefficients*. For instance, an application coefficient would be 1.0 if a subdomain expert were definitely applicable, and 0.0 if inapplicable, and smoothly vary between those limits when approaching the "boundary" of applicability.

SNAC seems like a solution to our problem, but there is a technical complication: SNAC requires that all subdomain experts have the same units, so that application coefficients can "mix" their evaluations at appropriate weights. How would you mix point differential with bingo probability? How would you even estimate bingo probability? Because it is hard to compare apples and oranges, I term this difficulty the "Fruit Basket Effect."

It is possible (or even probable) that there is a way to finesse this particular issue, and maybe even a few others. But complicating an evaluation function runs the risk of degrading its performance. In practice, it can be better to stick to one overarching theme, and do that one thing as well as possible. In MAVEN, the one thing is point differential. MAVEN is willing to make sacrifices as long as it can keep a high point differential. When this results in a weakness, as when defensive plays are best, MAVEN accepts the downside because there is a practical benefit to keeping the design pure. Chapter 10 will describe how simplicity works out for the best in the end.

To return to the subject of defense, there are techniques besides blocking and scoring that help maintain the lead. These techniques are so subtle that it is hard to recognize them as defensive play, yet they can be just as effective as scoring and blocking.

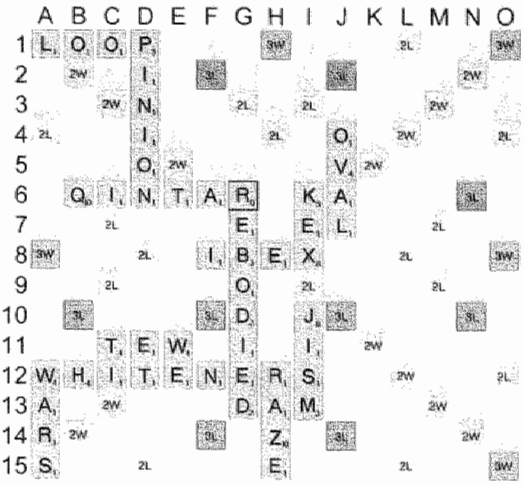
Simply turning over many tiles is beneficial because it shortens the game. If the opponent has fewer turns, then he will lay down fewer bingos. MAVEN is good at turning tiles, so it plays this strategy well.

Another technique applies to wide-open middle game positions. It is hopeless to try to block the bingo lines, since in the middle game bingo lines are like the heads of the Hydra; cut one off and another will grow in its place. It pays to view the problem from the perspective of minimizing the opponent's chances over the whole game. You can actually reduce the opponent's chances by giving him more bingo lines, because if he cannot use them immediately then the board will be filled up. The point is that a board is not truly closed until every quadrant contains tiles. In the middle game, the defensive goal is not to block the board, but to *fill it up*. MAVEN is good at filling the board, but it does not do it deliberately.

Blocking can be the best strategy, but it is difficult to find such moves without doing a simulation. The interaction of many factors, including the availability of alternative strategies, makes defending a lead a complicated decision.

Analysis and experience have convinced me that the opportunities for mechanically blocking the board are limited to the last few turns (i.e., the pre-endgame) and that in the rest of the game most humans overdo blocking to the detriment of their winning chances. An example is in order.

Nick Ballard had Position 6-3 in a game, and then published the position in *Medleys* [36]. The strategic situation is that the opponent is down 77 points, so he probably needs two bingos to win the game. The expert panel spread their votes over nine moves, as in Table 6-1. These options pale, however, beside the truly brilliant move played in the game—a move not even considered by the expert panel. Nick Ballard played "EULOGISM" (4G, 74). This is definitely the play, if you can get away with it.



Position 6-3 Many Defensive Tactics

Word	Spot	Score	Leave	Comments
VUG	5J	14	EILMS	A fish for a killing bingo.
KALIUM	6I	18	EMS	Higher score & turnover than VUG. Big S-hook.
MOGUL	4I	16	EIS	Good leave. Fills top quadrant. Nice S-hook.
REGULI	14A	18	MS	Turnover and a partial block of the bottom row.
EMU	J13	18	GILS	Shuts row 15 and the K-column.
SMUG	K6	19	EIL	Blocks K-column. Scores well, but burns the S.
MOIL	5C	22	EGSU	High score with a good-but-not-great rack leave.
VELUM	5J	20	GIS	Like KALIUM + higher score and weaker leave.
UM	2C	20	EGILS	Like VUG. More points, weaker leave.

Table 6-1 Best Plays for Position 6-3

In Brian Cappelletto's [23] view, the solution is to prevent the opponent from bingoing. By making one bingo a little harder, we substantially cut the chance of two bingos. For example, cutting the opponent's chance of bingoing from 1/7 to 1/8 would drop his chance of getting two bingos from 15% to 12%. Cappelletto was convinced that EMU (J13) was obvious, with no other choices worth a thought.

Peter Morris's[24] view was that it was too early to try to close down the board, because the opponent could keep it open. After all, there are still plenty of tiles in the bag, including a blank. Morris figures that the goal should not be to close the board, but to fill it up. Accordingly, he likes KALIUM because it starts to make plays in the upper left.

[23] World and North American Champion.

[24] Also World and North American Champion, which tells you a lot about the diversity of opinion among Scrabble experts.

A move like REGULI has high turnover. The idea here is that by turning over tiles we can end the game more quickly. This is valuable because the chance of 2 bingos in 5 turns (at 1/7 chance per turn) is 15.2%, but the chance of 2 bingos in 4 turns is only 10.0%. Of course, REGULI does not take a whole turn off the game. To take a whole turn off the game requires 8.4 tiles of turnover, a feat that is seldom accomplished in a single move.

Then there is VUG, which has the highest point differential of all plays. The attitude here is, "You lay down your plays, I will lay down mine, and when the game over we will find out who won." VUG is a fish for a killing bingo.

There is still another point of view. MOIL scores 22, the point being that racking up points makes it harder for the opponent to catch up.

Which play is best? Obviously "EULOGISM." Ballard said that his opponent did not even blink at the phony. Analysis, including simulations to the end of the game, suggests that the mechanical block of EMU is premature, and that MOGUL is the best play. MOGUL is not the extreme play from any perspective, but it does many things well. It scores a little more than most plays, turns over an extra tile, blocks a bingo line, creates a hook spot for our S, but the hook spot is not in the O-column so we do not risk losing our lead in one turn, it fills up one sector of the board and gets a start on the upper right quadrant. In short, a move that should make everyone happy, right? Well, actually, nobody but the author thought MOGUL was best, though everyone thought MOGUL was decent.

Obviously there are many ways to defend a lead. All of the perspectives given here can be the most important, depending on the situation. MAVEN's actual design plays to maximize point differential until the pre-endgame, and then introduces mechanical blocking into the mix. Chapter 9 outlines the technique.

6.2.9 Skepticism Sets In

These experiences showed four things. The first was to treat expert guidance with skepticism. Experts often do not know *which* move is best, let alone *why* it is best. We resolved to rely primarily upon rigorous testing rather than expert guidance.

The second lesson was that humans overestimate the significance of positional factors. A later section will explain why positional factors are generally small.

Third, the author learned that even when a positional theory had some validity it was still difficult to implement in the context of all of the tradeoffs, limitations, contingencies, probabilities and suppositions. Incomplete implementations of valid theories were worse than having nothing at all. Humans, too, suffered from misconceptions having the nature of "implementation errors," often choosing inappropriate moves even when they had unlimited time for pondering.

6.2.10 Historical Significance

Humans have historically overestimated the significance of positional factors. Consider, for example, the following advice from a Web site [37]:

> "Play the board not the rack - don't go chasing bonuses and forget the state of the game."

Do not listen to this advice!! If you take this advice literally, it will absolutely destroy your game. Far better is to ignore the board completely. Moreover, *definitely* chase bonuses (which is Aussie for "fish for bingos").

In particular, humans overestimate the impact of defensive considerations. Players try to "shut down the board" as soon as they get the lead, as if having the opponent play a bingo was the only way they could lose. In competitions, MAVEN has won games despite being out-bingoed, simply because MAVEN's other moves were not self-defeating.

Echoing a common reaction of early adopters of MAVEN, Joel Wapnick said that at first he thought MAVEN would suffer from its risky style of play. However, time after time he found that he could not punish MAVEN, and then to make matters worse, sometimes MAVEN punished him! This is exactly what theory predicts. Wapnick eventually concluded that MAVEN's style was actually not as risky as it seemed.

MAVEN legitimized the style of playing wide-open games. Besides being stylish and exciting, this style actually results in the greatest tournament success.

6.3 Generally Neutral

MAVEN has used 0.0 as the positional value since its inception. At first, this was because its author did not understand Scrabble theory. Now that the author has investigated the issues in some depth, we find that 0.0 is actually a good approximation! There are three reasons why positional evaluations (except for rack leave) are generally neutral.

First, you must subtract off the average score of all plays from the average score of a hotspot. For example, if the opponent scores 40 points in a spot, it does not mean that the spot cost 40 points. If the opponent's average score is 35 points, then the net loss is just 5 points, which is not a big deal considering that you are not sacrificing points to block the spot.

Second, creating a spot does not matter unless it is the *biggest* spot. If MAVEN creates the second-biggest spot and the opponent takes the biggest, then MAVEN benefits.

Third, the board is shared between MAVEN and the opponent. Maybe the opponent benefits, but maybe no one does, and maybe you do. Granted, the opponent does move first, so the opponent's chances are higher, but how much higher? Suppose that there are 11 moves left in the game (which there are, on average) so the opponent takes a particular hotspot 6 times, and you take it 5 times. What is the net deficit, on average? *One-eleventh* of the extra points gained. It is insignificant.

Thus, qualitative and quantitative arguments substantiate the initial theory that the board is neutral. Exceptional situations are characterized by synergy between your own rack

leave and the board (an exception because your rack leave is private, so this is a "one-way" opportunity) or a spot on the board that the opponent will usually take for a high score. Few such exceptional situations exist, and only one that I know of is easy to characterize. That one exception is the subject of the next section.

6.4 Triple Word Squares

Triple Word Squares are exceptional because it is easy to score 30+ points by covering one such square. Moreover, most racks contain the tiles needed to exploit the opening, so the opponent is highly likely to benefit. It is so easy to obtain a big score that this is the only exception to the general rule that board factors are irrelevant. MAVEN has a table of parameters that determine how damaging it is to open a triple word square as a function of the "shape" of the opening.

The penalties range from 0 (e.g., for putting a Q on A14) to 12 points (e.g., for putting an E on 1E when both 1A and 1H are open). The highest penalty is for placing an E adjacent to the DLS. This makes sense. Not only is the triple-triple chance high, but an opponent merely needs a single heavy tile to score 36+ points, and it is possible to hit 72+ with a Z or X in hand. The lowest penalty is for a Q in the seventh position, which has a penalty of 0. When you think about it, it makes sense; the only words that could use the spot are SUQS and TRANQS, but these words were not in the dictionary when this table debuted.

6.5 Bingo Openness

MAVEN incorporates a measure of bingo openness that encapsulates significant understanding of bingo chances. This section describes how it works. The key element is the quantity and quality of bingo lines. A bingo line is defined as a place where a seven or eight letter word can go on the board without interference from unnecessary tiles. We will break up the evaluation of bingo lines into rules for seven-letter lines and for eight-letter lines.

6.5.1 Evaluating Lines for Eight-Letter Bingos

An eight-letter bingo line consists of eight consecutive squares, of which exactly one is occupied, and having no interference from the rest of the board. The location and identity of the occupied tile is the crucial issue. A line consisting of ???????S is relatively likely to be used for a bingo, whereas a line ???????Q cannot be used for a bingo.

As our proxy for the value of a bingo line, then, we take the number of words that fit the pattern. For ???????S there are 9663 words that would fit. For ???????Q there are zero words. If we add up this figure over all possible bingo lines then we get a measure that correlates well with bingo probabilities.

6.5.2 Evaluating Lines for Seven-Letter Bingos

Seven-letter lines are a little different. A seven-letter bingo line consists of seven empty squares including one hook square. The letters that hook are crucial. For instance, if BDEGHLMNRSTWXY are the valid hook tiles and the pattern is *??????, where * stands for the hook square, then the chance of a bingo is relatively large. If the only hook tile is A and the pattern is ??????*, then the chance is small.

To maintain comparability with the eight-letter evaluation, we count how many words have a hook letter in the correct relative location. Any rack containing the tiles from such a word will make a bingo. However, you must account for the depletion of the bag. For instance, if the bag has no B's, then the availability of bingos starting with B is not helpful. So we prorate the word counts according to the relative density of the tile in the bag. An example is in order. Suppose that there are 40 tiles remaining, including one D, which is a hook tile. Suppose that 1000 words have a D in the right spot for the bingo line. Suppose that the bag has only 1 D in 40 tiles (a 1/40 rate) whereas the full bag has 4 D's in 100 tiles (a 1/25 rate). In this case, we would prorate the 1000 words by 25/40 to reflect D's frequency in this bag. Note that it is possible for prorating to increase the weight of a hook tile. The final step for evaluating a seven-letter bingo line is to add up the prorated weights.

6.5.3 Putting it Together

To evaluate the openness of a board, we add up the weights for all bingo lines. We then adjust this by a factor that reflects the impact of the blanks. A higher than average number of blanks increases bingos, which in MAVEN's model is a multiplicative adjustment. If you want to get sophisticated then you can adjust for the availability of every type of tile. Such adjustments are likely to be small for tiles other than blanks.

The measure is a proxy for the receptiveness of the board to bingos. You can convert this to a probability by using logarithmic regression. Actually, if you want to convert this into a probability then you should separately consider the board and the tiles, and use multiple logarithmic regression.

What can you do with this measure? The first thought is to use it as an evaluation feature, but unfortunately, it does not explain any of the variance in point differential. That is, it is a useless feature; as noted elsewhere, there is no general adjustment for bingo openness.

MAVEN uses this measure as an input to a neural network that estimates the chance of winning a game. The dominant factor in winning chances is the difference in score, of course. The next largest factor is the variance in scoring, of which bingo chance is a major component. The impact is indirect, but worth considering since it is inexpensive.

6.6 Winning Percentage Evaluation

You may want to know the chance of winning from a position. The best estimate comes from simulating the game to the end for an ungodly number of trials. Chapter 10 covers simulation, but all you need to know for this section is that simulation involves playing out games using random racks.

For example, suppose that a position is about even, and you want to know the actual winning percentage. The standard deviation of 2500 trials is $\sqrt{(2500 * 0.5 * 0.5)} = 25$. Therefore, the estimate of the mean would have a standard deviation of 25/2500 = 1%. If all you wanted to know were, "about how often will I win?" then this is good enough. If you wanted to know "does DICES or DEICES win more often?" then this might not be good enough. In addition, simulations to the end of a game take a long time: up to 20 moves per trial.

Is there a faster way to get an approximate answer? Can the method produce accurate answers when used within shallow simulations? The answer is "Yes" to both questions. A neural network can provide reasonable evaluations of winning chances. The estimates are not as accurate as those produced by long simulations, but they are good enough for practical purposes. It is quick to evaluate, which means that you can add such computations to shallow simulations without compromising throughput.

Tesauro pioneered the use of neural networks for evaluating backgammon positions [38]. His procedure was to use self-play to generate training instances, and use temporal difference training to evolve the neural network. This method should do well in Scrabble, too, but there are procedures that are more effective because the Scrabble space has two key resources that are not available in backgammon.

First, we have the MAVEN engine to supply training examples. In backgammon, the quality of play evolves along with the network, which starts out making arbitrary moves and learns from experience how to distinguish good positions from bad. It is good that backgammon programs can bootstrap in this way, but it causes long training times. In Scrabble, we have an inexhaustible supply of training games whose quality does not vary with the evolution of the network.

Second, each Scrabble position generalizes to many training instances for a neural network because the score is an independent, separable component of the position. Let us take an example. Suppose you play out a position and observe that the side to move gains 75 points over the rest of the game. If that player were 75 points down at the start then the game is a tie, if he is 74 or fewer points down then he wins, and if 76 or more points down then he loses. MAVEN uses one training position to train the neural network at several scores. One training score is the score that ties the game, and then we train the network using other scores that are symmetrically placed about the tying score.

There are several advantages to this method. First, training is faster because we do not need to recompute all of the inputs for every training instance. Second, the resulting network is sensitive to changes in the score input, which is important because it is the dominant contributor to winning chances. Third, the network never has positions where it thinks that increasing the score is bad owing to statistical noise in training; in this procedure, the training instances always force higher scores to have higher evaluations regardless of the other input settings.

The result is good. An informal comparison of the network estimates with actual games suggests that the standard error is small, on the order of 3%. This is accurate enough for roughly evaluating how often a position wins. It is not obvious that it is sufficiently accurate for choosing good moves, but it definitely is. You might think that if plays were randomly misevaluated by 3% then the wrong move would often be selected. However, the errors committed by a neural network when evaluating the descendants of a given position are highly correlated. For example, suppose the network consistently overestimates certain positions. If the network has to choose a move in such a position, then the network overestimates all of the moves, and therefore it selects the best one. On an emotional level, you hate the thought that your program consistently makes an error, but in practice that is far superior to making random errors.

The network is useful in conjunction with shallow simulations for making accurate evaluations of winning chances. The procedure is to simulate random variations to a fixed depth. For example, suppose that you simulate to a fixed depth of 3 ply. Such a simulation will run much more quickly than simulating to the end of the game, which would involve 21-ply variations. In addition to completing each iteration 7 times faster, there is lower variance. The variance of a sequence of 21 moves is 7 times the variance of a sequence of 3 moves, so the simulation converges $\sqrt{7}$ times faster.

The preceding analysis shows that truncated simulations converge faster than simulating to the end of the game. Is it problematic that correlated errors occur in neural network evaluations? In theory it is. In practice, making three moves from the root produces a wide variety of positions, so errors in endpoint evaluation are likely to cancel.

6.7 Evaluating Endgames

An endgame is a position in which all of the tiles have been drawn from the bag, and therefore the game is deterministic. Evaluating endgame positions quickly and accurately has an important role in the algorithms for using simulations (Chapter 10) to estimate winning percentage. This section describes a process that evaluates endgames.

In Chapter 9, you will find that MAVEN has an endgame search engine that can play out an endgame with extraordinary accuracy. MAVEN could just play out an endgame and see what the result is. However, that would take time. In particular, it is too slow to use in simulations. Is there a procedure that returns an evaluation of an endgame position without requiring a deep search?

First, let us qualify what we mean by "fast and accurate enough." The goal is to surpass two alternatives. One alternative is to evaluate endgames by playing them out using the endgame search engine. The other alternative is to play endgames out using MAVEN's midgame evaluator, with tile values tuned for endgame play.

The search engine is accurate. Almost perfect, actually. We estimate its standard error at less than 1 point per evaluation. Unfortunately, the rate of evaluations per second is only about 1. The speed is also highly variable, since individual moves may take much longer.

Evaluation by playing the endgame out using the midgame evaluator is possible. For selecting moves in endgames, we would change the rack evaluation parameters. The method actually used is to set the evaluation parameters to -2 times the face value of the tile, plus a bonus of double the opponent's rack if a moves uses all of the tiles. This evaluation is correct if we can go out in one move, or if the opponent is guaranteed to go out in one. Moreover, is reasonable for longer sequences as well.

Such a rack evaluator can select moves in the endgame at great speed. A typical endgame lasts for 3 moves, but the last one is unusually fast because the racks are almost empty. It is best to think of the speed as 2 move generations. Thus, this process can evaluate endgames at a rate exceeding 100 per second. The speed is consistent, too.

The accuracy of the general evaluator is poor, however. The estimate has a standard error of 13 points. Such an error rate is too large, since about 9% of all games are decided by 13 points or less. Moreover, it is in the close games that you really need accurate

endgame evaluation, so this evaluator hurts most when it is most needed. In addition to the large standard error, there is the possibility that errors can be huge. For example, if the second player to move threatens a bingo out, then the first player will predict a variation such as Play for 30, Bingo for -80 minus tiles, which has a net of about -60. However, the actual game will proceed as follows: Block for 15, Play for -30, Play Out for 18 plus tiles, which has a net of about +10. The total error is 70 points![25] The implication is that even a large predicted lead is not truly safe.

Now we can qualify what it means to be fast and accurate. We want to compute the estimate using only a few move generations. And we want a standard error well under 13 points.

MAVEN contains such an evaluation function. It costs the equivalent of about 4 move generations, so it runs at one-half the speed of the midgame analyzer. Its computational speed is essentially constant, which is an advantage over the endgame search engine. Its standard error is 5 points, so it is much more accurate than the midgame analyzer.

The endgame evaluator has two large advantages over using the midgame analyzer. One advantage is important when the game is close. Only 4% of all games end with a point differential less than one standard error, compared with 9% when using the midgame evaluator.

The second advantage is that the analyzer returns an estimate of the winning chance. For instance, if the endgame evaluator predicts an 8-point margin of victory, then experience shows that this corresponds to a 96% chance of victory. This is an improvement for two reasons. First, the midgame analyzer would simply predict a win (i.e., 100% chance of victory), which is inaccurate given a 13-point standard error. Second, if you have extra time and want to be really sure about which variations actually win, you can apply the endgame search engine to the positions that are too close to call, using the evaluator's winning percentage estimate to prioritize the variations. Only a small fraction of all positions will have uncertain winning percentages, so you can get essentially perfect evaluations while using few endgame searches.

The second advantage helps when the game is not close. The evaluator is virtually never in error by 15 points, so when a predicted point differential exceeds 15 points then the prediction is certain. Only 10% of all games end with differentials less than 15 points, so uncertainty in winning percentage is lower. In addition, as described above, the endgame search engine can be applied to the undecided games if time allows, with the computation performed in priority order.

There is a human dimension to judging close endgames accurately. When using the midgame analyzer to play out endgames, a 1-point loss is recorded as 0% chance of a win, whereas when the endgame evaluator sees a 1-point loss it will rate the position as a 47.8% chance of a win. The impact of this difference is to encourage the creation of more close endgames. Since MAVEN plays the endgame much more accurately than humans do, this is to MAVEN's advantage. If MAVEN obtains an endgame that a human has to play perfectly, MAVEN will chalk up a win more often than not.

---

[25] Refer to the end of the game CRAB-TYLER, from Chapter 3.

The endgame evaluator works by generating the plays using the endgame search engine's move generator. That generator returns each move with a set of evaluation bounds, Low and High. The bounds can be wide, and are subject to all sorts of suppositions regarding the future of the endgame. Those suppositions are tested and validated in the endgame search engine. However, in the endgame evaluator there is no time for search, so we must predict the value using largely static analysis (i.e., using no additional full-board move generations).

The evaluation process is a tree search, but the constraint is that all of the moves are pre-generated. (Well, we generate setups off of the tiles of the side-to-move on the first turn, but this incremental generation is smaller than a full-board generation.) Within that constraint, we conduct a tree search that aims to narrow the bounds of the search. In outline, then the algorithm is the following:

1) Generate many plausible moves.
2) Search the endgame as if these were the only legal moves.
3) Well, sometimes we will generate setups, but only when necessary.

This process is less accurate than the endgame player, but still accurate. Difficulties arise when the move generation process is obviously insufficient. For example, suppose we determine that the opponent is stuck with a tile. Then the variations will often involve little one-tile plays and small setups. The opponent will have little flexibility. However, the static view of the tree is limiting because key moves may not have been saved for incorporation into the tree. There is a chicken-and-egg problem; we must generate plays and analyze them before we discover that the opponent is stuck. If we prioritize moves by immediate scoring value then we are unlikely to keep one-tile plays. The heuristic we employ in that case is to guess near the High bound, and we are right more often than not.

Another case is to save several moves for the opponent's tiles, and then discover that they were all blocked. Odds are excellent that the opponent has some moves, but we have not saved enough of them. We resolve this by guessing a residual value close to the upper bound. Sometimes we guess that we missed a key move for the opponent. For example, consider a play that has a lot of uncertainty because we are unsure whether we can play our follow-up play. For example, we threaten to go out, but we have only one such move. Will that move be blocked? Sometimes we know for sure that it will, because among the opponent's plays is a blocking move that seems best. Suppose that the opponent should block, but we just have not saved such a play? Then we must specifically generate moves that block the follow-up play, and consider how they work out.

Ultimately the tree search concludes with narrowed bounds for the true value. Then we have to pick a point inside. Whether the score tends to work out at the high end or at the low end depends on several factors, the most significant of which is the side to move at the end of the variation we are predicting. Whoever moves at the end of the variation will have first crack at improving the score.

Finally, we compute a winning percentage by looking up the predicted point differential in a table. Actually there are several tables, according to the length of the predicted variation. Short variations have low standard error. For example, consider the extremes of variation length. A variation of length 1 means that the side-to-move bingoed out, and such moves are always best (in practice) and the evaluation is always accurate. A

variation of length 2 means that the opponent played out on his turn, and the search engine would only allow such a thing if it were basically forced, so there is low error. At the other extreme, when one player is stuck, then the evaluation involves many suppositions, and the error rates are highest.

Please note that we could have computed winning percentages in the same way if we continued using the midgame analyzer. That is, we could use the predicted point differential as an index into a table that predicts winning percentage. However, because of the frequency of bingo outs, that table would have a lot of uncertainty. For example, in a variation of length 2, should we assume that the opponent's out play was unblockable? The difference between blockable and unblockable could be huge, and the average case estimate may be misleading. Additionally, the reason for using a winning percentage estimate is that the winning percentage is a nonlinear function of the point differential. For the endgame evaluator that is normally true; that curve has a sharp rise between point differentials of -5 to +5. When the curve spreads out because of a higher standard error, and is distorted by errors exceeding 70 points, then it is not clear that the curve will have the desired effect.

The endgame evaluation function meets the goals of acceptable speed and accuracy. Simulations are half as fast as when using the general analyzer for the same purpose, but it pays off by cutting the standard error by 60%.

The value of reducing the error rate is evident when we consider MAVEN's ability to evaluate moves that empty the bag. Before implementing this evaluator, MAVEN could compute exact winning percentages (i.e., exact with respect to the assumed probability distribution of opponent's racks) for situations with one tile in the bag only. The reason was that the evaluation of the 8 possible endgames was an expensive proposition because the endgame search engine was called on each one. To extend that to two tiles in the bag would have involved analyzing up to 36 possible endgames. It may be feasible using today's faster computers.

However, the endgame evaluator substantially raises the bar. If a move empties the bag with 5 tiles remaining then MAVEN can compute an almost perfect estimate of the winning chances. With 5 tiles remaining there are up to 792 endgames to evaluate. Each involves the equivalent of 4 move generations, so the total cost is 3960 move generations. Expensive, perhaps, but suddenly feasible. Accuracy will be practically perfect in those cases that have a point differential greater than 15. In the remaining cases (usually less than 10%) the endgame search engine can improve the analysis as time allows or until the move is either proven or eliminated.

This evaluator significantly raises the accuracy of simulations. However, there is a hidden downside. (There is always a downside.) That is the classic two-player game error of presuming that the opponent will see what the program sees, which I will dub the "Mirror Effect." There may be times when the only chance of victory is to hope for an opponent's error. This is illustrated by the next example.

Position 6-4 occurred in the 1992 National Championship. The author reviewed the play as a member of the committee responsible for awarding the tournament's Brilliancy Prize. In submitting this position, the player to move noted

> "Since I was down 64 points with the rack ABGGIU?, and with one tile in the bag (ELNNORUW unseen), I knew that I needed a bingo to have any chance to win. But what should I draw for? My opponent was tile tracking, so I knew that if I played for an S bingo (i.e. BUGGIES, BAGWIGS, etc.), he could simply block with ON, EN, OR or ER 15J or other, better plays through the E at F13. However, if I use the B and draw the R for ZIGGURAT B6, he might not notice it."

Position 6-4 Fantastic Fish for a Swindle

The player realized that he has no chance if his opponent played perfectly. Indeed, against MAVEN he could resign himself to defeat. However, against a human, there is a chance. This player did not give up, but continued to calculate his best moves. (Recall that Chapter 4 mentioned that humans are amazing.) What happened in the game is a classic example of the good luck that comes to those who prepare for it. Here is the story, as told by the winner:

> "Ironically, after playing BE 10N 10pt., I drew the R, the opponent saw ZIGGURAT and mistakenly played EWE C6 to block, allowing the formerly unplayable ARGUING B8, 76pt., winning."

This is a classic example of how winners make their own luck. Hats off to the victor! It turned out that the lucky winner was Joe Edley. Brilliancy Prize submissions are anonymous, so that player reputations are not considered when the prize is awarded. The opponent was Randy Hersom, who finished in the top 10 of this event.

Against MAVEN, Edley's move would have no chance. Edley's move emptied the bag, leaving an endgame that MAVEN would analyze with certainty. Alas, because of the Mirror Effect, if MAVEN were in Edley's seat it would not fish, because it would assume that Hersom would see ZIGGURAT and block.

It is possible that if MAVEN were using the general analyzer to play out endgames then it would fish for ZIGGURAT. If it generated Edley's BE it would then enter an endgame with Hersom moving and ZIGGURAT pending in 1/8 of all cases. Whether MAVEN's move would accidentally block ZIGGURAT (or inadvertently set up a different bingo) is

unknown. So there are many uncertainties. If you want to generate swindles then you need to be aware of the Mirror Effect and take specific calculated risks to counteract it.

It is an open question whether to exploit human errors, and how. See Chapter 12 for a list of possibilities.

Chapter 7 – Early Game Play

The early game covers that part of the game in which the score of the game is not a significant factor in the move selection process. Normally the early game covers about 7 moves for each side. The play of computers is truly dominating in the early game. Computerized evaluation functions correlate well with the quality of moves, so the program's exhaustive move generation is a formidable asset.

This section gives the selection criterion for choosing moves in the early game, and then gives several examples of computer play. We finish with a summary of the characteristics of computer play in the early game.

7.1 Selection Criterion

MAVEN chooses moves according to its estimate of point differential. In particular, MAVEN does not directly estimate winning chances. This is a closer approximation than you might guess. The justification for choosing moves based upon point differential is that the difference between the scores of the players over the rest of a long game is approximately normally distributed. Therefore,

winning percentage = Cumulative Normal ((Our Score – Opp Score) / √variance)

Thus, the point differential is directly related to winning percentage. If a move has lower point differential it must have a difference in variance large enough to compensate. However, the variance over the rest of a long game is hard to influence. You might be able to affect the variance of the next turn or two, but the board will evolve and the variance is likely to return to normal, or at any rate, is likely to be about the same across the immediate successors of a root position. Reducing the variance on this turn has little impact on the variance of the game as a whole.

Exceptional situations have few turns left. Even there, only rarely does it transpire that the best move has a lower point differential than another play. In the early game, the best move almost always has the highest point differential.

7.2 Characteristics of MAVEN's Early Game Strategy

In the early game, MAVEN is hard to beat. Errors are usually minor. Errors occur because MAVEN misevaluates a move. Often the board or rack is extreme in some way, and the positional averages do not apply exactly. Usually the move selected by MAVEN is good anyway.

It is useful to single out positions where the best move falls outside of the top 10 moves in the ordering. That case is interesting because it means that MAVEN is missing some substantial concept. In the current implementation, such plays are usually fishing plays.

Consider the opening rack ADEHORT. MAVEN is content to take 28 points by playing THREAD. Fishing is better. OH (8H) scores only 10, but it compensates by taking points from the opponent and by increasing chances for landing a bingo. Table 7-1 shows simulation results.

Word	Square	Simulated Differential
OH	8H	26.6
THREAD	8C	22.4
HATRED	8D	21.9

Table 7-1 MAVEN Plays THREAD instead of OH

As you can see from simulation results, OH is better than THREAD by about 4 points. Overlooking fishing moves like OH happens regularly, and is a significant loss in total.

But as a practical matter, MAVEN benefits from opening the board. MAVEN never misses the bingos that occur after THREAD, and human opponents do. It is probably still correct to play OH, but MAVEN's style of play offers definite compensation for missing occasional fishes.

David Gibson submitted the opening rack AFNNORT. Gibson played FRONT (8D,24,NR) in the game, which is the move MAVEN would play. He noted the alternatives NONFAT, FANON, and FON. Gibson wondered if FON should get the edge, because of defense (fewer bingo lines), rack leave (ANRT is better than AR). The question is Do these advantages outweigh the downside of FON, which scores 12 points less than FRONT?

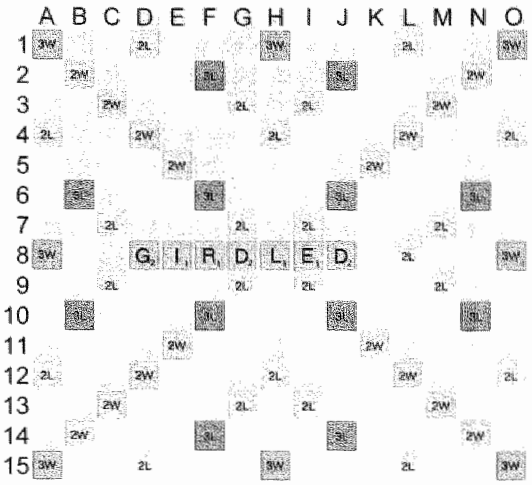
Word	Square	Score	Rack	Simulated Differential
FRONT	8D	24	AN	26.8
FANON	8D	24	RT	26.2
NONFAT	8D	20	R	23.7
FON	8G	12	ANRT	21.4
FONT	8G	14	ANR	20.1
FANO	8G	14	NRT	17.9
FAN	8G	12	NORT	17.6

Table 7-2 Example Showing that Fishing is Often Wrong

A MAVEN simulation collected the data in Table 7-2. The data show that FON regains value, but too little to catch FRONT. The bingo synergy and defensive value of FON do not overcome a 12-point sacrifice. FRONT is superior to FON by 5.4 points. Since FON sacrifices 12 points, but finishes only 5.4 points back in equity, it must have advantages that regain 12 - 5.4 = 6.6 points. Most of this difference is due to defensive factors.

FRONT vs FON shows that bingo synergy does not necessarily accrue to racks that have low and non-duplicate tiles. There are other qualities, which are often not obvious by inspection.

In Position 7-1, Joe Edley bingoed to start the game, and Robert Felt held drek. Felt played KEV (I7, 19, GNSTT).[26] Edley was aghast. He was certain that exchanging was best. Nevertheless, Felt stuck to his play, and with good reason. Exchanging (keeping ST, most likely) would have a net value of about 7 or 8 points under the Basic model, and KEV's Basic evaluation is about 10.



Position 7-1 Example of Typical Small Error

For a second opinion, Edley called Nick Ballard, who spotted TSKING (E5, 22, TV). Edley and Ballard were certain that TSKING was best, and MAVEN's Basic model concurs. The Basic model sees that KEV scores 3 fewer points, and the GNNST rack leave is consonant-laden and has a duplicated N. There is some compensation from the S, but overall GNNST seems pretty bad. The NV rack leave after TSKING is also bad, but at least that rack has only one negative (the V) so there is hope in the draw from the bag. For GNNST, by contrast, there are 3 negatives (G, NN, 5 consonants) against 1 positive (the S) and the draw is only two tiles so not much can happen. I recall that Bob preferred KEV over TSKING, which seemed like pure stubbornness in light of the analysis of score and rack leave.

Table 7-3 shows the Basic assessment of the two moves. MAVEN's Basic model regards TSKING to be 7 points superior to KEV. In the early game, such a margin is usually proof of superiority. However, this situation is a little peculiar. MAVEN simulations showed *KEV* was best. KEV actually gets its points back, and Felt was right all along.

Word	Leave	Score	BasicI	V/C	Total
TSKING	NV	22	–6.5	+1.5	17
KEV	GNNST	19	–3.0	–6.5	10

Table 7-3 Basic Evaluation of KEV and TSKING

Simulations show that the GNSTT leave bingos 19.6% of the time, usually through the I in GIRDLED, making an ING ending. After TSKING, the rack is NV, and we get a bingo only 11.3% of the time. KEV also has a defensive superiority, in that the opponent's bingo frequency drops from 22.8% to 18.9%.

[26] KEV was in the OSPD3, but not valid in TWL98. I have always wondered how that happens. Does a lexicographer wake up one day and say, "Hey! That's not a word! Who put that in here?"

The total edge in bingo scoring over the next two turns is 19.6% - 11.3% + 22.8% - 18.9% times 50 points (the average difference between bingo and non-bingo moves) which equals 6.1 points. This almost makes up the difference by itself.

KEV has about a point edge over TSKING when all is said and done. Thus, MAVEN makes an error but it is not particularly significant.

7.3 Overall Assessment of Early Game Play

The play of the Basic model in the early game is remarkably robust. The moves are usually correct. When a move is wrong, the error is usually small. Besides, many errors pay stylistic dividends owing to the exploitable weakness of humans in open positions.

Chapter 8 – Endgame Play

The endgame is a game of perfect information. That is because each player can subtract the known tiles (i.e., those on the player's rack and those on the board) from the fixed initial distribution to arrive at the set of tiles that the opponent must have. It was surprising to find out that human experts do this routinely, a process known as *tile tracking*. Some players even allocate their time so that they have 10 minutes (out of 25) for thinking through the endgame and pre-endgame moves.

What characterizes good endgame play? Well, scoring is important, as always, but not at the expense of going out, which means using all of your tiles, thereby ending the game. Going out first is important since if your opponent goes out then you lose points in three ways: your opponent gets an extra turn, you do not receive his tile penalty, and he obtains your tile penalty.

There are defensive aspects to endgame play. Here are three examples. First, if your opponent has one big hotspot then you want to block. Second, if your opponent has just one place to play a specific tile, then you want to take that opportunity away. Third, if your opponent cannot play out because he is stuck with a tile, then you want to maximize the value of your tiles by playing out slowly.

What sort of algorithm leads to good endgame moves? Let us start by investigating the obvious. Full-width alpha-beta search is the conventional technique for perfect-information two-player games. However, it works badly in Scrabble for many reasons.

1) The branching factor is rather large: about 200 for a 7-tile versus 7-tile endgame.

2) Search depth can be an issue. When one player is stuck with the Q, it may take 14 ply to justify the correct sequence, which is often to play off one tile at a time.

3) Alpha-beta's performance depends critically on good move ordering. Unfortunately, Scrabble move generators produce moves in prescribed orders. Of course, you can generate the moves and then sort them, but a good general-purpose heuristic for scoring moves is not obvious. For example, in stuck-with-Q endgames the best move is often low scoring.

4) Move generation in Scrabble is computationally expensive. The author considers MAVEN to be fast, but MAVEN is much too slow to perform a real-time 14-ply full-width search.

Therefore, we rejected full-width alpha-beta search. However, a personal communication from John Chew indicated that he and James Cherry were investigating Scrabble endgames by adding a full-width component to ACBOT. Perhaps alpha-beta will stage a comeback using faster computers and clever adaptations to the domain. We have doubts, since branching factors of 200 and depths of 14 ply are not so easily dismissed. Indeed, at that time Chew indicated that ACBOT's endgame engine was not useable during games, because it took too long. More recent communication suggests that ACBot has a capable real-time endgame player based upon alpha-beta [39].

To highlight the profundity of endgames we provide an example where one player is stuck with the Z. Position 8-1 occurred in a game between Joe Edley and Paul Avrin, with Edley to move [40]. Edley's rack was IKLMTZ and Avrin's was ?AEINRU.[27] Edley wishes to set up his Z, which is unplayable in the position now.

Position 8-1 Deep Endgame Example

Edley's move, TSK (4L, 18), is strongest. It threatens to follow with ZITI (L2, 13). Table 8-1 shows the most straightforward plan for the defender .

To surpass the game continuation requires a truly deep assessment of the position. Avrin missed Edley's setup of the Z, but his move is pretty strong nonetheless. If Avrin had seen the threat then he might have considered three blocking moves. The block that MAVEN selected was URN (2J, 3, ?AEIU). Two blocks that Joel Wapnick pointed out are UN (2J, 2, ?AEIRU) and TIE (L4, 8, ?AINRU). Justifying any of these three moves is difficult, because there is an immediate, tangible loss from blocking: Avrin will not be able to take the TWS using DRIVEN, and to make matters worse, Edley will take the TWS using VIM. This is a huge initial investment, and the gain from sticking Edley with the Z is, by itself, insufficient compensation. To justify the block, Avrin must manufacture many more points than he can score with DRIVEN.

Player	Word	Spot	Score
Edley	TSK	4L	18
Avrin	DRIVEN	8J	-36
Edley	ZITI	L2	13
Avrin	AUK	N2	-14
Edley	(LM)		-8 = -27

Table 8-1 Game Moves of Position 8-1

MAVEN's play was URN. It seems to me that MAVEN's variation (which is given below) shows how Avrin can get more than 8 points for the E, so TIE is probably not best. But Wapnick's suggestion of UN is intriguing, since it preserves an R that can score decently (e.g., PAR (2E, 7) or BEER (4D, 6)). Someday the author will have to investigate why URN was selected instead of UN. Perhaps URN is actually better because of an obscure subtlety. Another factor to consider is that MAVEN's endgame search would have been terminated by a timeout, so the search may have stopped having proven only that URN was a good move, but without a proof that it was best. I ran this case over 10 years ago on a 20 MHz computer, so it is likely that modern hardware would yield different results.

[27] It is unusual to see an endgame with 6 tiles moving against 7. The only way this can happen is if the side with 7 tiles played zero tiles on his previous turn. In this game, Avrin lost his previous turn when Edley challenged a phony.

The tactics that justify blocking ZITI are surprisingly deep. MAVEN quotes the variation of Table 8-2 as best. The author suspects that Wapnick's UN is better. If UN is the best play, then the variation for UN is probably similar.

Player	Word	Spot	Score	Commentary
Edley	TSK	4L	18	Best for offense. What should defense do?
Avrin	URN	2J	-3	Blocks ZITI (L2), but that allows...
Edley	VIM	8M	24	Edley to get the triple-word square.
Avrin	MU	O8	-4	Avrin plays out one time at a time
Edley			0	
Avrin	ENJOYS	I9	-16	Can Avrin get back enough points?
Edley			0	
Avrin	IN	12K	-4	What is Avrin getting at?
Edley	PAL	2E	7	Avrin threatens to go out, so Edley plays.
Avrin	VIATICA	5H	-18	Oh!
Edley			0	No play for the Z, of course.
Avrin	HE	6M	-16	Oh!! Avrin gets 34 points on his last moves!
Edley	(Z)		-20 = -32	

Table 8-2 Best Play Variation for Position 8-1

This variation, 12 moves long, need not be seen in its entirety in order to choose TSK. TSK may be selected after noting that it is high scoring and sets up a strong threat to counter the opponent's play DRIVEN.

By contrast, to choose URN requires seeing the entire 11-move sequence that follows, since only the stunning setup VIATICA followed by HE justifies the initial sacrifices of points. Note that the opponent passes to delay VIATICA, because it can be followed by VIATICAL. The defender finally forces his opponent to play off the L with PAL, and then VIATICA and HE follow.

8.1 First Try

We have previously described MAVEN's first effort, which was to naively look ahead at the top N moves using a forward search. This system should have improved MAVEN, but a buggy implementation failed. Upon reflection, we decided that this approach was doomed, because

1) blocking the opponent happened largely as a matter of chance,

2) there was no incentive to play one tile at a time when the opponent was stuck,

3) finding plans that went out in two moves was unreliable, and

4) depth of search was only two plies.

In short, there was no underlying plan that guided move selection, so the top N moves were likely to miss key resources. Therefore, MAVEN started over from scratch.

8.2 Domain Analysis and Requirements

An endgame is a perfect-information game, with a high branching factor and potentially a requirement for deep search. Nevertheless, humans manage to find strong moves, and it is worth investigating how they do this.

One helpful property is that Scrabble is a *converging game*. That is, the complexity declines along every variation as tiles are placed onto the board. Thus, it is possible that an analysis made at the root of the search tree will turn up information that is valid (or at least useful) throughout the search.

A second attribute is that the board is largely fixed during the endgame. An endgame may add up to 13 tiles to the board, but there are already at least 86 tiles on the board when it starts. That the board is largely frozen supports the notion that a static analysis of the root will be useful in the search.

A third trait is that the outcome of a variation is the sum of the scores of the branches (with the opponent's scores taken as having a negative sign). Moreover, the number of moves added together is normally small, and often only 3. It follows that simply choosing high scores is a key tactic.

Nevertheless, playing out quickly must be paramount. Allowing your opponent to play out before you do is too painful to contemplate. Letting the opponent get an extra move, using tiles that you could have taken as a penalty, and then having to pay him a tile penalty often adds up to over 20 points, and it can be worse than that. Therefore, the key goal is to play out as quickly as possible, while maximizing the sum of the moves.

Since you start with a rack of tiles and never add any tiles to that rack, you can envision how you might plan to play off your own tiles using dynamic programming. Specifically, generate all moves, and then devise a plan that combines moves that eventually use all tiles. You can even extend this to compute the best plan that is N moves long, for every value of N from 0 to 7. Now, how do you handle the opponent's moves?

One consideration must be to reduce the opponent's score. We have already mentioned that playing out quickly is a vitally important component of the strategy. There is additionally the move-by-move tactics of blocking the opponent's high-scoring plays. How would you achieve that goal?

If you generated a sorted list of the opponent's high-scoring moves then you could scan that list for the highest-ranking move that is not blocked by each of your moves. Actually, that description is a little misleading, because which of the opponent's moves is optimal depends on the tiles you keep. In reality, you must consider many opposing plays and pick the one that plays best against the tiles you keep.

So the idea, then, is to form a plan through static analysis, then validate it using search.

These notions sketch a solution, but there is more to do. Obviously, the generators are not perfect, and we must embed them in a search somehow, and so on. Still, the basic notion is established: the solution will compute an *oracle* that is specially tuned to this position.

That oracle is expected to be reasonably expert for evaluating the entire space underneath the root, since the entire space differs from the root relatively little.

8.3 Evaluation Function

Since we have rejected exhaustive search, we must necessarily be selective, which implies the need for an accurate evaluation function. We can construct such a function because Scrabble endgames are converging and largely static, as described in the previous section.

These properties are exploited by using dynamic programming. Because endgame positions are largely static, the evaluation depends largely on the two racks that the players hold. In other words, if the position reduces to one where we hold ER and our opponent holds AW, then the evaluation of that position is largely determined. It might vary as a function of changes made to the board position, but we can deal with that in the search engine.

It is impractical to build a table of all possible racks that we hold versus all possible racks that our opponent holds, as there are up to 128 possible racks for each side. Fortunately, we do not need the full set. Simply knowing what one side can achieve within N turns (for N = 0, ..., 7) is sufficient.

For instance, suppose that the side to move can play out in two turns, and the opponent cannot play out in one turn. Then a good estimate of the endgame is the largest possible two-turn plan that plays out for the side to move minus the best one-turn plan for the opponent.

Of course, the best two-turn plan for the side-to-move depends on many things. It depends on the whole position, actually. But the board is largely static. There are already at least 86 tiles on the board, and the few that will be added during the endgame are only a small fraction of the total. It follows that the legal moves that are possible from a given rack are largely fixed.

The opponent's rack may have an impact on the best plan. In particular, the opponent will try to resist, so you can expect the best plan to be somewhat worse than the best plan in the absence of opposition. While this is a complicating factor, it does suggest that searching moves in order of the value of the "opposition-less" plan will work well. It is true that the evaluation is not perfect, but the intention is to place the evaluation within a search engine, which should be able to overcome the defects.

These considerations suggest that an excellent approximation to the value of an endgame is a plan computed under the assumption that there is no opposition and the board never changes. Specifically, for each value of N = 0, ..., 7, and for each rack that a player may hold in this endgame[28], we wish to compute the largest sum of N turns that can be made from that rack, and set a flag indicating whether the N turns play out all tiles.

[28] There are no more than 128 possible racks that each player may hold, since tiles are never introduced into play once the endgame starts.

We can statically compute the values of the racks if N=0 (i.e., if the opponent goes out before we move). That is simply –2 times the sum of the tiles. For N > 0 we use dynamic programming. For every move that plays out of each rack, we compute the score of that move plus the evaluation of the remaining tiles assuming that N-1 turns remain. The value of a rack in N turns is the largest value of that function over all possible moves.

Constructing the evaluation table for a player requires just one full-board move generation, since the computation assumes that the set of legal moves is static. The power of the evaluation function computed by dynamic programming is that it *approximates the results of a deep search*, and the beauty of it is that it is *cheap to compute*.

We can then do an evaluation of any position in the search space by using a type of quiescence search [41]. In a classical quiescence search from chess programming theory, the player to move at a frontier node as the choice between "standing pat" (which ends the variation) or trying to improve his score by generating captures. A similar process occurs in Scrabble endgames, where the player has the option between playing out and playing on. Normally it is best to play out as soon as possible, but when the opponent's tiles are bad then it can pay to play on instead, compounding the opponent's difficulties. We have seen such a case when Joe Edley was stuck with a Z, and it was in Paul Avrin's favor to play out slowly. What Avrin would be thinking on his turn is, "Should I play out now, or should I try for more?" That question is exactly a quiescence search. MAVEN includes such a search in its endgame player.

For example, suppose that MAVEN needs to evaluate a position where the tiles DERTTT move first against the tiles EJVY. If DERTTT can play out, then that establishes a lower bound on the value of the variation. Otherwise, DERTTT follows its best plan, and then EJVY has the chance to play. The process repeats with EJVY to move. Since the number of tiles decreases monotonically, eventually the process ends.

The previous paragraph describes the process as a search, but because the options are all static and the search "tree" never has branching factor > 1, it is possible to write a loop that computes the same result. The loop actually computes the value "bottom up" by figuring out first what will happen if the side to move is allowed to make 8 turns (and the opponent makes 7). Then the opponent is given the option of playing out in 7 turns, and so on. At each iteration the players try to improve upon their fate by playing out sooner.

This evaluation function is static and fast. It takes two full-board move generations to construct the evaluation tables. We can then do an evaluation of any position in the search space.

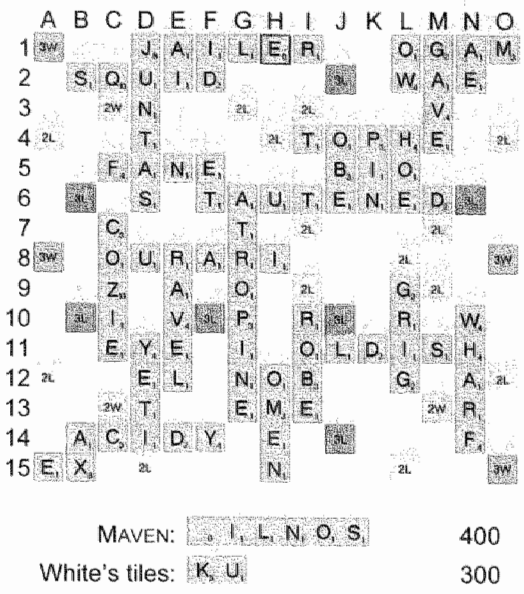
8.4 Second Search Engine

Our first successful endgame player did a two-ply search of moves ordered by this evaluation function. This program was good, because it could execute the key skills. It scored points, because it included all the high-scoring plays. It could block, if any blocking play was included among the top N moves (which is usually true; hotspots for the opponent are usually at least somewhat hot for us). It could play out in two moves because the evaluation of positions after 2 plies includes the observation that the tiles we keep will be out in one. It could play out one tile at a time if the opponent is stuck, because it would see that it scores more points in seven turns than in two turns.

Position 8-2 is from a December 1987 tournament game, with MAVEN moving versus Charlene White. White had only one way to use the K on her rack: PINK (K4, 10). MAVEN noticed this and blocked. There were a few ways to block PINK, but two moves stand out: PINON (K4, 7) and OBELI (J4, 7). The first is stronger; the one-tiling value of IL exceeds that of NO because of LAX (B13). The whole variation is in Table 8-3.

Word	Spot	Score
PINON	K4	+7
EMEU	13G	-6
COZIES	C7	+20
WHARFS	1D	+14
LAX	B13	+10
IT	4C	+7
(K)		+10 = 62

Table 8-3 Best Play for Position 8-2



Position 8-2 MAVEN's First Sharp Endgame Play

Prior to having an endgame player, MAVEN would never find a plan like this. MAVEN would make a move like NOISE (15K, 30, L). White would reply with PINK (K4, 10), and MAVEN would play out with LAX (B13, 10+2). That dreadful sequence nets only 22, so MAVEN's error would cost 40 points. PINON is well worth finding, as it is the equivalent of scoring an extra bingo.

8.5 Weakness of Second Engine

The engine was not perfect, but it was a big step forward. We suspected that it was better than human experts were, and indeed, it was a huge advance for computer programs. Unfortunately, the approach had at least four deficiencies that were hard to address without extensions to the framework.

1) Blocking occurred largely as a matter of chance.

2) The engine found setups largely as a matter of chance.

3) *Negative setups*, where the opponent can use MAVEN's tiles beneficially, unpleasantly surprised the engine.

4) As it stood, the algorithm spent the same amount of time on every play.

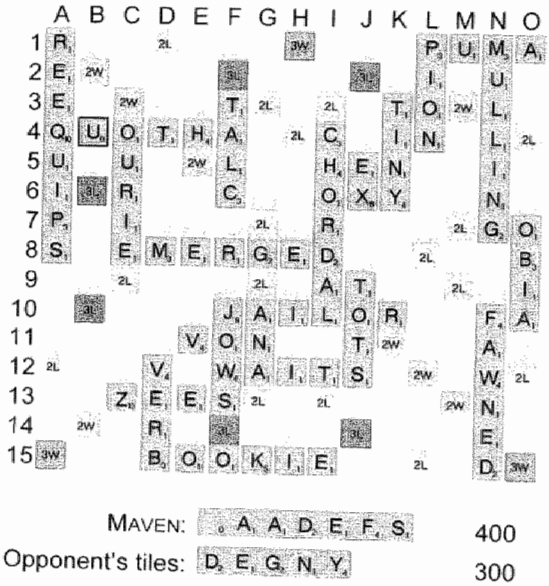
Part of the problem was that the plausible-move generator did not take into account the opponent's possibilities. That was easy to fix, since we were already generating the opponent's plays while building the evaluation tables. We simply saved a set of

promising opposing moves. Then, after generating moves for the side-to-move we could determine which of these opposing moves were not blocked, and score the resulting variations after a two-move sequence was played.

This tweak was a big improvement, but there remained shortcomings that suggested that the search did not faithfully model the domain. For example, suppose the opponent holds ROADWAY, and he can play ROAD for a small score, and WAY for a big score. Actually, WAY is the opponent's only big score, and we must block it. However, when we evaluate a blocking move, we find that the evaluation may be inaccurate. For example, among the opponent's replies we find ROAD. If we are not out-in-one after our initial move then the evaluation of the future is "way off," because the evaluation tables believe that holding WAY is valuable. The change in position is not reflected in the evaluation tables until the opponent generates plausible moves.

To generalize this example, the plan constructed by the evaluation function maximizes the score achieved by the sum of the moves, but two factors are unspecified. One factor is the order of the moves; the moves of the plan can be played in any order. It is up to the search engine to determine whether any of our moves interfere with one another. Another factor unspecified by the plan is the amount of risk—what happens if a move is not playable? These issues are delegated to the search engine, but the search engine fails to accept the responsibility.

The problem afflicted MAVEN for a while. We considered modifying the evaluation function to account for the block of WAY, but that would just introduce other problems. For example, suppose that WAY could be played in two locations, one scoring 25 points and the other 28 points. The gain from blocking the high-scoring spot is only 3 points. How would the evaluation function know what the gain was?



Position 8-3 Blocking, Going Out in Two

A different deficiency was that the engine had no ability to foresee resources deeper in the position, because the static analysis was only done once. Position 8-3 shows what can happen. This position is from an October 1987 tournament, before MAVEN had any endgame player. The move actually played was FASHED (L8, 40), the highest-scoring play.

The plan is to score well using the -S hook at 10L, while interfering with the opponent's EDDY (15L, 30) and DEWY (12L, 30). A secondary priority is scoring highly with FA

(O14, 29).[29] Thus, MAVEN likes the plays from L10 using ?ADES (keeping AF) that begin with S. There are many such plays — SABED, SADES, SANED, SATED, SAVED, SAWED, SADHE, SEDAN, SHADE, SPADE, SPAED, STADE, and STEAD — and it is unclear which one works.

SEDAN is best (see Table 8-4) because other plays allow the opponent to block FA (O14). The words ending in ED—SABED, SANED, SATED, SAVED, SAWED, and SPAED—are refuted by EYNE (13L, 14), which nets 35 after FAD (14J, 15). EYNE also refutes SADES, but with a net of 33. SHADE, SPADE, and STADE are refuted by DYNE (13L, 16). STEAD is refuted by DYED (15K, 26). After SEDAN, the opponent can block FA, but only at a price: after YENNED (14J, 17) the opponent is stuck with the G! Then FOXY (6H, 17) and DA (15N, 18) are unstoppable, and the net is 46. To see these complexities requires that the position be reanalyzed at every node.

Best	Spot	Score
SEDAN	L10	+24
EDGY	2A	-18
FA	O14	+29
(EN)		4 = 39

Actual	Spot	Score
FASHED	L8	+40
EDDY	15L	-30
AT	3E	+7
(GN)		6 = 23

Table 8-4 Variations from Position 8-3

8.6 B* Search Algorithm

The classic approach to a search engine that fails to accept responsibility is *massive overkill*. That is, simply search more moves more deeply. It is true that this is a reliable way of riding the technology curve. In effect, the search engine can delegate the problem to Intel.

Nowadays that solution would stand a good chance of working. However, the computers of 1990 were not so fast, and it did not seem like overkill would prevail any time soon. It must also be considered that overkill applies just as well to clever algorithms, so the author set about trying to find one.

The analysis of how MAVEN's ad hoc algorithm failed lit the path to a solution. The real issue is the difference between the highest and second-highest score using a set of tiles. Accordingly, the evaluation function generalized to including an analysis of the *risk* associated with an N-turn sequence. In effect, there was an optimistic evaluation and a pessimistic evaluation for every variation. The idea was that the true evaluation would be within those bounds. Computing such bounds is tricky, but with sufficient test cases you can eventually make a system that usually works.

At that point MAVEN had an evaluation function that faithfully modeled the domain. It remained to find a search engine that worked with evaluations expressed as intervals. The answer is Hans Berliner's B* algorithm [19]. The B* algorithm is an admissible[30] search algorithm that addresses single-agent or adversarial search problems. In Scrabble, we use the adversarial formulation.

---

[29] FA creates the crossword DA, which was removed from TWL98 when someone realized that it is a foreign word that occurs only in names.

[30] That is to say that it always finds the optimal solution if certain conditions are met. In B*'s case the conditions include unlimited time, unlimited space, and an evaluation function that returns an interval that always contains the true value of a position.

The outermost loop of B* is to select a frontier node of the search tree (which must be kept in memory) for "expansion," which means that the static evaluation of the selected node is replaced by a one-ply search of its successors. Changes in the evaluation of that node are "backed up" through the search tree using a process akin to minimax. The outermost loop terminates when a stopping criterion is met. The details of the implementation of these steps distinguish B* from other in-memory algorithms having the same basic selective-extension framework. The rest of this section will describe the algorithm in detail.

The first distinctive feature of B* is that the evaluation of nodes is an interval. The algorithm assumes that the interval necessarily contains the true value of the node, but the example of MAVEN demonstrates that B* is fairly robust to occasional violations of this condition. The lower bound of an evaluation is called the Pessimistic score, and the upper bound is called the Optimistic score. MAVEN uses a "negamax"[31] formulation of the update rule for backing up scores through the tree, so the update rule takes the form

$$\text{Optimistic(Parent)} = \text{Max}(-\text{Pessimistic(Children)})$$
$$\text{Pessimistic(Parent)} = \text{Max}(-\text{Optimistic(Children)}),$$

where the Max operation is taken over all children. Note the negation that is characteristic of negamax. Notice how intervals are minimaxed: by setting the parent's optimistic bound to its child's pessimistic bound. Another peculiarity of B* is that the optimistic and pessimistic bounds need not come from the same child. One of the strengths of B* is that it can identify cases where one child provides a secure pessimistic bound, while another node provides upside.

The criterion for stopping any in-memory search engine must include practical criteria related to resource exhaustion, and MAVEN follows in this practice. However, such limits are seldom invoked because the search process usually terminates because of a criterion that is unique to B*: *separation*. A B* search achieves separation when the pessimistic score of one move is at least as large as the optimistic score of all other moves. Under the assumption that the true value of every node is always contained within the interval, the separation condition amounts to a proof that one move is at least as good as any other.

There is one last decision to make in a B* algorithm: which node should be expanded? B* provides two strategies, called Prove-Best and Disprove-Rest, for making this decision. The goal of Prove-Best is to raise the pessimistic bound of the move that has the highest optimistic bound. If the pessimistic bound of that move rises enough, then separation will be achieved. The goal of Disprove-Rest is to lower the optimistic bound of the move having the second-highest optimistic bound. The reader can verify that there is no point in working on changing the bounds of any other moves, since it is impossible to achieve separation until progress is made on at least one of these two goals.

If the strategy is Prove-Best, then the frontier node to expand is found by following the sequence of nodes that establish optimistic bounds for the root node with the highest optimistic bound until a frontier node is found. If the strategy selection is Disprove-Rest,

[31] In a negamax formulation of a minimax tree, the value of a node is always measured with respect to the side to move in that node. Thus, high scores are always good for the side to move. When you back up a negamax score, you always have to negate the value from level to level.

then the node is found by tracing the second-highest optimistic bound. It follows that one of at most two nodes is the node to expand. But which should be selected?

It is clear that Prove-Best should be selected until the node having the highest optimistic score is also the node having the highest pessimistic score. Until that condition applies, it is impossible for Disprove-Rest to achieve separation even if it is completely successful. Once that level of proof is achieved, Disprove-Rest is a viable strategy, and sometimes it is faster to work on disproving moves than to work on proving moves. This can happen, for example, if the tree under the best optimistic move is large. Such a tree might have a large "conspiracy number,"[32] [42] whereas smaller trees under other moves might be easily solved.

It is also obvious that Prove-Best should not be selected if its optimistic bound equals its pessimistic bound, since then there is nothing to learn. With the easy cases resolved, we turn to the complex case where there is still uncertainty in both the biggest and second biggest moves, and the intervals overlap. What heuristic should we use to choose a strategy?

Berliner's original paper proposed a few heuristics for selecting the strategy, and later papers by Palay [43] and Berliner and McConnell [44] consider the issue in elaborate detail. But MAVEN does not use any of these because they seemed unwieldy. In keeping with MAVEN's engineering-oriented implementation philosophy, MAVEN adopts a simple strategy that cannot be fooled. MAVEN alternates between Prove-Best and Disprove-Rest. This guarantees that MAVEN will search any position with at most twice the minimum possible number of node expansions.

B* works like a charm. In part, this is because the evaluation function and plausible-move generator are tremendously accurate. In part, this is because the domain is converging and shallow. We know of only one other domain [33] where B* is superior to all other known techniques.

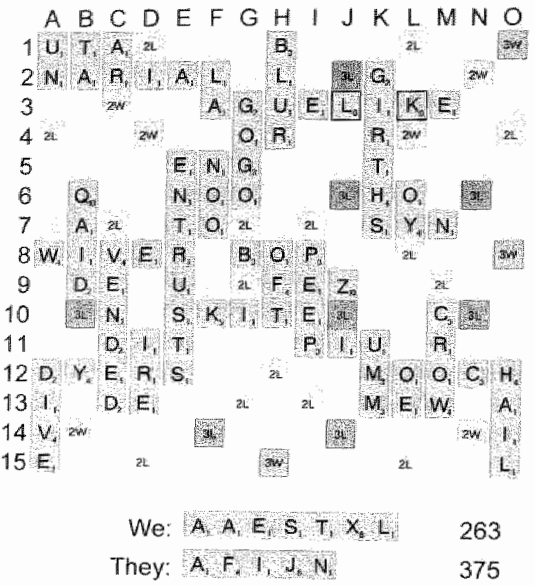
[32] A tree's conspiracy number equals the smallest number of nodes whose values must change in order to propagate a different value to the root of the tree.

8.7 Execution Trace

We will use Position 8-4 to illustrate the progression of B* as it searches.

When the engine examines this position, the first thing it notices is that the opponent has few threats. The opponent's highest scoring play is JAIL (15L, 19, FN), but it has the drawback of keeping only FN, and the only play for FN is FEN (M2, 6). It follows that if the opponent plays JAIL then he is at risk of being unable to play out in two moves.

In situations like this, the real challenge is to find plays that do not improve the opponent's prospects. This is hard to do with a rack like AAESTXL, because these tiles are easy to overlap. It follows that we may need to try several moves to determine which creates the least counterplay.



Position 8-4 Endgame execution trace

The best play on first glance is AXELS (N5, 35, AT). AXELS scores well, keeps an out-in-two, and also interferes with several moves the opponent might like to make, such as FIN (3M, 12) and RANI (3K, 12). The evaluation function bounds on AXELS are [39,67], meaning that MAVEN believes that a net of 39 points is guaranteed, and the total may reach as high as 67.

You might wonder how 67 points could possibly be achieved. With full insight into the dynamics of the position we know that only 40 points can be achieved, but a static view sees additional potential. On the defensive side, MAVEN sees that AXELS interferes with several moves that the opponent would like to use to play an out-in-two. MAVEN thinks it is possible that AXELS will force the opponent to go out-in-three, which could be a big win. Additionally, AXELS sets up TAN (M6, 19), making the crossword HOAX with the X played by AXELS. This is a 5-point upgrade over the biggest play for AT prior to AXELS, which is AT (J5, 14). Accordingly, MAVEN upgrades the optimistic bound for AXELS.

The optimism proves to be a false hope when MAVEN expands AXELS. The opponent replies FAN (M6, 25, JIN), scoring well and also blocking both TAN (M6), and AT (J5). The opponent's JIN threatens to play out at O3, so there is no possibility of one-tiling the opponent. MAVEN accepts that the variation AXELS, FAN, followed by TA (N14, 10+20) is forced, with a net of 40.

The new high move is AXEL (N5, 34, AST), with a similar theme, but this time the AST rack leave promises more punch if the opponent cannot play out in two. MAVEN expands AXEL, and sees the opponent's reply JAIL (15L, 19, FN). The opponent threatens to play

out with FEN (M2, 6). There is no profitable way to block FEN using AST, so MAVEN accepts that AXEL, JAIL, followed by SAT (J13, 14+10) is forced, for a net of 39.

The next high move is SAT (J13, 14). MAVEN sees that AXEL (N5, 34) is a guaranteed out play, and MAVEN threatens LATEX (15H, 60). This gives SAT a static evaluation of [25,65] at first. When SAT is expanded, we discover that the opponent can use the 15th row to play FAINT (15H, 24). Actually, MAVEN saw this possibility when statically analyzing SAT, but the node expansion gave MAVEN the chance to see if FAINT allows any comeback shots that improve results. The biggest potential is that FAINT keeps a bare J, which would often have trouble playing out, raising the hope of a one-tile attack. Alas, the opponent threatens TAJ (B1, 10). Despite the fact that TAJ is blockable, this is sufficient to forestall a one-tile attack. MAVEN accepts that the best-play variation is SAT, FAINT, AXEL (N5, 34+16), which nets 40.

MAVEN's next expansion is interesting: SHOAT (6J, 14, AELX). The motivation for SHOAT is that it keeps AELX, which threatens AXEL (N5). What MAVEN has not noticed until now is that SHOAT blocks AXEL, so the entire theory is doomed. Of course, B* is meant to address situations like this. The expansion discovers that AELX can only play out with LATEX (N4, 14), and the variation concludes with a bound of [4,36], which effectively excludes SHOAT from further consideration because we already have a guarantee of 40.

The sixth expansion is LAXEST (N4, 36, A). This move was initially ranked behind AXELS because AXELS created larger setup possibilities, but those have proven illusory so LAXEST now has the biggest upside. MAVEN sees that DITA (11C, 11) and the newly created HOAX (6K, 16) imply a forced out play, and it remains to see if the opponent's setup of FAN (M5, 25) is as valuable as it seems. In fact, the best play variation is LAXEST, FAN, DITA, for a net of 42. Thus, LAXEST achieves the highest pessimistic bound.

There are still moves with higher optimistic bounds, so MAVEN continues using the Prove-Best strategy. It tries TAS (12G, 13, AELX), a move with the same idea as SAT (J13), but lacking any setup threats. TAS proves insufficient, and the variation terminates with a bound of [3,38].

MAVEN next tries the idea HOAX (6K, 32). The point is that by playing HOAX rather than LAXEST you prevent the opponent from playing FAN in reply. Moreover, the AELST on the rack should have a good chance against AFIJN, even if AFIJN moves first. Expansion shows that AELST poses two threats: LAXEST (N4, 20) and SETAL (14D, 11). The low score of the latter play may doom HOAX if the opponent can block LAXEST with a good score. If the opponent does not block then HOAX nets 43 after JAIL (15L, 19) and LAXEST. But the opponent has blocking plays (FAN (4M, 12, AJ) and FIN (4M, 12, IJ)) that are wilder, and MAVEN could only place a lower bound of 24 on these. So this expansion terminates with a range of [24,43]. This interval straddles the net of 42 achieved by LAXEST, so the interval must be narrowed further to achieve separation. But before we apply the Disprove-Rest strategy to HOAX, we must eliminate a few other moves with higher optimistic scores.

A nice out-in-two threat is SEX (14D, 30, AALT), followed by RATAL (4K, 16) next turn. Unfortunately, the opponent's natural moves (JEAN(M2), FIN (4M), and FAN

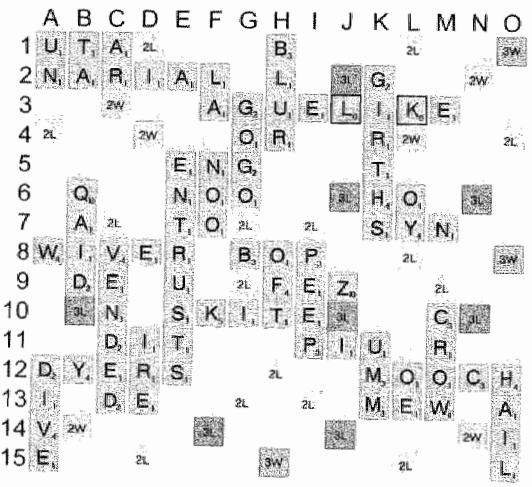
(4M)) interfere with RATAL, and do not generate setup opportunities. So this variation fails to go out in two moves, and earns an evaluation of [9,26].

A better out-in-two threat is SAX (14D, 30, AELT), because it leaves two threats to go out: LATE (N4, 13) and RATEL (4K, 16). Unfortunately, the opponent can simply play JAIL (15L, 19, FN). Then SAX plus RATEL is 46, minus JAIL plus FN makes a net of only 37. The opponent might do even better with FIN (3M) or FAN (3M), which block both out plays.

Another out-in-two threat is RAX (3K, 31, AELST). This play scores well and interferes with many of the opponent's plays. Unfortunately, the threats to go out are not great; SETAL (14D) scores only 11, and STEAL (N5) scores only 14. So the opponent can play JAIL (15L, 19, FN), after which STEAL nets only 36. To defeat JAIL you must exploit the fact that FN cannot play out because RAX blocks FEN (M2). MAVEN's evaluation function estimates the upside of the situation as 42 points. Thus, this expansion returns an interval of [36,42], which effectively eliminates RAX from contention.

A different plan is AT (J5, 14), keeping the tiles for AXELS (N5, 35). Note the pattern of trying high-scoring plays like AXELS first, then trying moves like AT that keep the tiles (e.g., AELSX) for high-scoring plays. This pattern is sensible, but it is not coded into the engine. It arises out of the Prove-Best strategy, which starts with the move having the highest optimistic bound (AXELS), then progresses to other moves having high upsides. The evaluation function sees that AT has a high upside because there is the potential of following with AXELS, so eventually the engine gets around to AT. When AT is expanded, alas, MAVEN discovers that AXELS can be blocked by FIN (3M) and FAN (3M), so the new bounds are only [11,40].

MAVEN now changes tack, by blocking JAIL with LA (N14, 10, AESTX). This keeps a threat of TAXES (N4, 35) as an out play. MAVEN's static estimate was that this position is bounded by [-15,44]. Expansion shows that the opponent should block TAXES with FIN (3M), FAN (3M), or JEAN (M2), and now the position is complicated again because there are no guaranteed out plays. MAVEN figures that the opponent should play FIN, followed by MAVEN's HOAX (6K, 32, EST), but can only issue a bound of [13,48] on that variation.



It might seem to the reader that there is a bug here, because the original estimate was that the score of LA was less than 44, and now MAVEN is raising its bound to 48. That is an interesting issue. From the perspective of internal consistency, it is not necessarily a bug, because any value in the range [13,44] would satisfy both the original and the refined estimates. A disagreement with previous estimates only arises

when the true value lies in the range [45,48]. Note that B* does not necessarily produce the wrong answer if the evaluations are wrong. For instance, if the value of LA were 48, the engine would discover that after one more expansion, and its developer would trumpet that discovery as evidence of the "robustness" of the algorithm. On the downside, if the true value of RAX were 43 instead of the estimated [36,42] then the wrong move will be selected, because RAX has been permanently eliminated. (Well, not quite. More on that later.)

To return to the trace, LA is still the move with the highest optimistic score, but its pessimistic score is less than the 42 achieved by LAXEST, so Prove-Best is again selected. We descend through LA and FIN (the move that established the optimistic bound for LA) and we find that HOAX does indeed achieve a value of 48. So the range of LA remains at [13,48] while we look at better moves for the opponent.

The next expansion is LA followed by FAN, which is better because the opponent now threatens JIN (A2, 10) playing out. The move that blocks JIN with the highest optimistic score is TA (5N, 6), which threatens to go out with SEX (14D, 30). But this is a small threat, and the net of the variation is only 35 at best. This reduces the bounds on LA to [17,35], which eliminates LA.

Another concept for blocking JAIL is TA (N14, 10), keeping AXELS. The original bounds were [30,44]. MAVEN expects FIN, FAN or JEAN after TA, with an evaluation in the range [6,46].

TA is again the biggest move, and we check TA followed by FIN first. But FIN is not a good move because it does not preserve an out play. The bounds change to [9,46], and TA is expanded again. This time the opponent plays FAN, and the threat of JIN (A2) out is refuting, with bounds of [17,30].

The biggest move is now HOAX, which was previously expanded and has bounds of [24,43]. The engine searches the reply IF (8M, 8) first. This looks dangerous, because IF runs up against the TWS at 8-O, but the danger is actually small because it is impossible to play out on the O-column owing to interference from HOAX and HAIL. Still, IF is a bad move because it does not preserve an out play, and afterwards the bounds on HOAX still stand at [24,43]. The next expansion is HOAX followed by FIN, and the same thing is discovered.

HOAX followed by FAN is the next expansion. MAVEN sees the threat of JIN (A2), and sees that it can play out with SETAL (14D, 11). This guarantees a value of 40 for HOAX, but the upper bound remains at 43 because MAVEN can try to block JIN with LEANT (O1, 15, S) instead, and this has potential to hit 43. So this expansion terminates with a bound of [40,43].

HOAX is still the highest optimistic score, so MAVEN looks at the expansion of HOAX and FAN again. MAVEN concludes that LEANT does not achieve a value of 43. If you want to block then you should play TA (4N, 16), with an optimistic bound of 40. This bounds HOAX at [40,40], so HOAX is eliminated.

At this point the highest optimistic score is 42, achieved by LAXEST and RAX. The highest pessimistic score is also 42, achieved by LAXEST. Strictly speaking, MAVEN

could stop at this point because RAX cannot surpass LAXEST if the evaluation function bounds are correct. But that is a dangerous assumption, so MAVEN does not stop at equality, but instead goes for strict separation. And that is fortunate, since otherwise you would not get to see the Disprove-Rest strategy.

MAVEN next expands RAX followed by JAIL, as selected by the Disprove-Rest strategy. The bounds of this alternative are [36,42] before the expansion. The expansion shows the variation SET (3A, 18, AL), FA (E1, 5, N), LA (2M, 14+2), for a net of 41. Uncertainties associated with the last two moves enlarge the interval to [38,47], which makes RAX the highest optimistic moves.

The Prove-Best strategy is the only applicable strategy now, so RAX is expanded, following the variation RAX, JAIL, and FEZES, which established the upper bound of 47. MAVEN now sees that ARF (C1, 12, N) followed by LAT (2M, 16+2) is forced, but the net is only 35. This narrows the range for RAX to [38,44].

The Prove-Best strategy descends to RAX, JAIL, and SET, which is strong because it blocks ARF. The variation FA (E1, 5, N) followed by LA (2M, 14+2) is forced, showing that the lower bound on RAX is 41. The upper bound is still 44, so RAX is selected by Prove-Best again.

The upper bound of 44 is suggested by the variation RAX, JAIL, and ET (3C, 12, ALS). ET is good because it blocks ARF. Again the opponent has nothing better than FA (E1, 5, N). So far so good. Compared with playing SET instead of ET, we have lost 6 points, but we now have an S in the rack leave. Can we make something of it? Actually, no. The biggest hook is the same whether the S is on the rack or not: SAL (2M, 16+2). So ET fails to improve, but that does reduce the optimistic bound of RAX to 43.

The final try is RAX followed by JAIL and SHOAL (6K, 14, ET). This plan had a high upper bound because ET plays for 13 points at 3B. However, the opponent's ARF (C1, 12, N) blocks ET, so this plan turns out to be a fantasy. This reduces the range of RAX to [41,41], which proves that LAXEST is the best play by a 1-point margin.

The efficiency of the process amazes me, even after years of watching it. This position, for instance, involves hundreds of legal moves per side, and the search considered variations that are 5 ply deep. Yet the engine achieved logical separation using only 27 nodes of search, which were completed in zero seconds on my 1GHz processor. On MAVEN's 1986 machine, this same search would have taken less than a minute, which is fast enough for tournament play.

8.8 Overall Evaluation

In 1990, the author conducted a study of endgame examples [45]. Cases came from published games and from endgame compositions. The study showed that MAVEN analyzes endgames much more accurately than humans do. In the game positions human experts averaged 10.5 points of errors per player per game, whereas MAVEN's error rate was close to zero. Moreover, MAVEN was more accurate than human annotators, who have the benefit of hindsight and the luxury of time for study. Annotators averaged 3.1 points in errors per game.

The only positions where humans outperformed MAVEN were the endgame compositions. Such positions were concocted to prove a point, rather than to match the distribution of endgame instances. MAVEN analyzed 21 out of 26 correctly, dropping 40 points, whereas human annotators analyzed 24 of 26 correctly, dropping 17 points.

Since then, there have been a few bug fixes, but nothing major. The author's enduring impression is that the endgame player is overwhelmingly accurate. The error rate may be less than one in a hundred games. It is hard to say because it is difficult to prove that any of MAVEN's moves is bad.

8.9 Still, it is not Perfect...

Here are examples of how MAVEN can go wrong. It does not go wrong often. Actually, almost all endgames are simple out-in-twos, which MAVEN nails every time. MAVEN also plays most complicated situations perfectly. Still, the complexities can confound even the most robust implementations.

Position 8-5 was printed in *Scrabble Players News*. The best play is SINE (12K, 13), which sets up WART (O12, 45). MAVEN plays SNAW (12l), a move having some tactical merit, but which nets 18 less than SINE. SINE would be a terrific find in an actual game. The opponent's tiles IINT are normally flexible enough to block such setups. Many players who find SINE rely on the score; if you are 45 points down then you need a big play. Anything that draws attention to relevant positional features is helpful.

Position 8-5 MAVEN Misses Endgame Setup

MAVEN cannot find setups for every legal move. We must draw the line somewhere because looking for setups is expensive; MAVEN already devotes 30% of its time to looking for setups.[33] It is unclear that increasing that fraction would improve MAVEN, because of the time subtracted from other useful methods.

Best	Spot	Score
SINE	12K	+13
NET	D8	-13
WART	O12	+45
(II)		+4 = 49

MAVEN	Spot	Score
SNAW	12L	+22
WIN	O12	-6 (!!)
RE	15N	+27
MITT	F1	-8
(IT)		-4 = 31

Table 8-5 Variations from Position 8-4

After MAVEN's SNAW, the opponent has a tough problem to solve, because MAVEN threatens TWIER (O15, 27 and out). The opponent's tiles are IINT, which cannot quite make a big play in the

[33] The rest goes to other move generation tasks.

O-column. The solution to the puzzle is clever (see Table 8-5) , and many players would miss it.

The author is evaluating several methods that find SINE, but do not risk an explosion in search time. Most promising is to exploit MAVEN's high opinion of SINEW (12K): MAVEN would do a setup analysis for SINE because:

1) The opponent cannot play out,

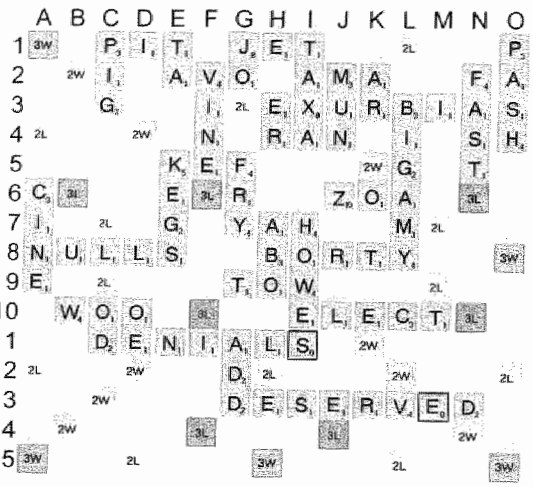
2) You can hook a tile to the end to form a word that is already on the best list, and

3) You can score highly using the hook location.

However, this process has limits. Position 8-6 shows how complicated the endgame can be. This position occurred between Rita Norr and Steven Alexander. Alexander has just played CINE (A6, 6), trying to prevent Norr from playing her Q (e.g., QUEEN (A4) threatened). Unfortunately, the Q still plays, as shown in Table 8-6.

MAVEN	Spot	Score
QUERCINE	A2	+20
IODIN	N11	-12
OVEN	L12	+16
(U)		+2 = 26

Table 8-6 Variation Found by MAVEN for Position 8-5



Position 8-6 Amazing Endgame Setup from Rita Norr

These moves, while surprising and effective, are not best. In the game, Norr missed QUERCINE, and we are lucky she did, since otherwise the beauty of this position would have been lost on us. Norr's crushing, devastating move was OD (N12, 3).

The key point of OD is that Alexander cannot block the setup of QUEEN or QUERN (O8, 48) without using his N, for example with MINI (7L, 7) or IODIN (N11, 12). However, after the N is gone, the deadly setup ER (H13, 2) creates an unblockable threat of QUEEN (15D, 81). It is actually best not

If Block	Spot	Score
OD	N12	+3
IODIN	N11	-12
ER	H13	+2
OM	7K	-7
QUEEN	15D	+81
(O)		+2 = 77

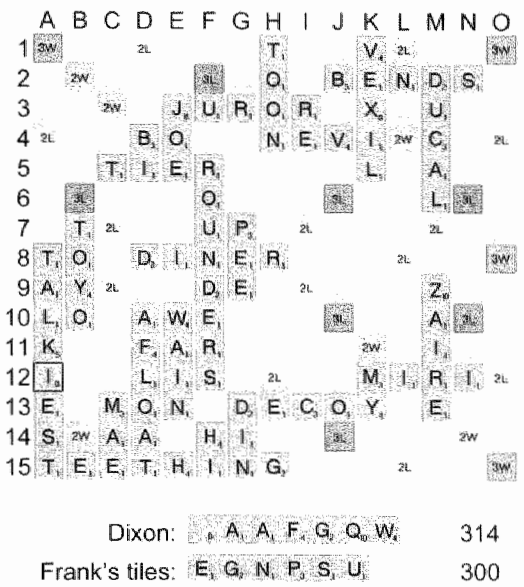
Best Play	Spot	Score
OD	N12	+3
UNDESERVED	13E	-14
QUERN	O8	+48
OM	7K	-7
DE	12G	+8
(II)		+4 = 42

Table 8-7 Variations from Position 8-5

to block OD, as Table 8-7 shows.

MAVEN actually finds the right moves for Alexander, correctly choosing not to block.[34] But MAVEN does not find OD for Norr. It is hard to see that OD is so crushing, so OD is cut off before any serious examination.

Finally, Position 8-7 is a tactically complex endgame. The truth is that MAVEN's move may be the best play in practice, because it is hard to see the refutation over the board. It is worth wading through the variations to appreciate both how MAVEN can err, and how often things go right despite the complexities.



Position 8-7 Tricky Endgame Example

Jan Dixon and Alan Frank played this endgame in 1987. Dixon is moving and leads by 14 points. Dixon has a blank but her other tiles are awkward, including the Q. The diagram has only one place to play the Q, through the A at 10M. Thus, play will proceed by either playing the Q there immediately, or making plays that set up other Q moves.

It is important to note that this position occurred before the introduction of QAT to the lexicon. Dixon's decision is much simpler if QAT is legal!

One thing to notice is that QUAG (10K, 17) is killed by UNPEGS (O5, 41 and out). Dixon must proceed more carefully.

The best play is FLAG (6L), which leads to a tied games after best play (Table 8-8). FLAG sets up QUA (7J) for the Q, while interfering with Frank's annoying –S hook onto VEXIL. The column labeled "Best" shows how Frank can tie the game by choosing the correct out-in-two. The column labeled "Bad Reply" shows that Frank can go wrong by choosing other, plausible out-in-twos.

Best	Spot	Score	Bad Reply	Spot	Score
FLAG	6I	+10	FLAG	6L	+10
GAEN	10L	-7	LEAP	5K	-15
QUA	7J	+16	AQUA	10K	+14
SUP	14M	-25	GNUS	7L	-13
(W)		-8 = -14	(W)		-8 = -12

Table 8-8 Best Moves for Position 8-6

[34] In the actual game, Alexander was not so fortunate. He played to block Norr. The extra 35 points that Norr scored were the difference between victory and defeat.

MAVEN does not select FLAG. MAVEN is beguiled by the tricky move WAT (C3, 17). WAT is a tremendous shot, which requires precise defense to defeat (Table 8-9). The defense is to keep SUEG to make SQUEG (1A) if Dixon plays QUA (B1). Best defense (left-hand column) shows that WAT has a net of –19 after best play. But it is easy to go wrong, as the defense recommended by Edley (right-hand columns) shows.

MAVEN	Spot	Score
WAT	C3	+17
PI	11M	-8
REF	I3	+6
HIN	14F	-10
VEXILS	K1	+15
BIG	D4	-6
ZAG	9M	+13
ZAIRES	M9	-15
US	6J	-3
OE	6F	-8
(Q)		-20 = -19

Bad Reply	Spot	Score
WAT	C3	+17
NAP	10M	-11
GAS	6I	+20
ECU	I12	-5
EF	13M	+10
BIG	D4	-6
ECUS	I12	-19
(Q)		-20 = -14

Table 8-9 A Tremendous Shot that Barely Fails

Note how, in the variation after NAP, the blank and FW score enough to hold off the useful and flexible tiles EGNPSU. Heavy tiles are not a disadvantage in one-tile endgames. MAVEN's ingenious attack against the defense NAP (10M) in the right-hand variation is based on blocking high-scoring plays, so that a one-tile battle is necessary. One challenge for Scrabble masters is to discover through the application of profound principles what MAVEN discovers by concrete calculation.

Edley found FLAW (6L), which seems clearly winning as the left-hand column of Table 8-10 seems obvious. But then he called back to say that he found a refutation, which turns out to be the same refutation that MAVEN found. The beautiful setup shot just barely defeats FLAW, as the right-hand column shows.

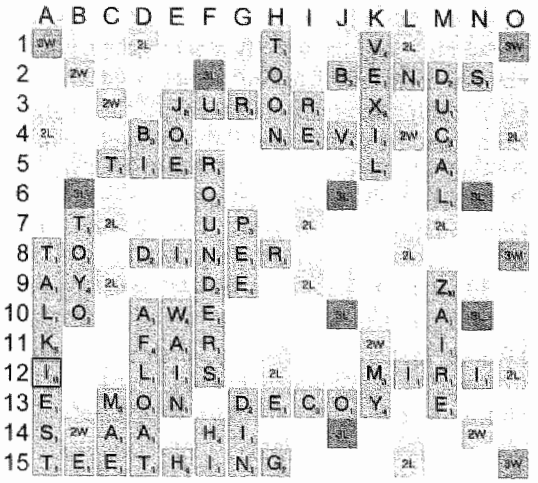
Flawed	Spot	Score
FLAW	6L	+12
SWUNG	O5	-27
QUAG	10L	+17
DEEP	9F	-12
(A)		-2 = -12

Flawless	Spot	Score
FLAW	6L	+12
LEAP	5K	-15 (!!)
WAG	O6	+21
AN	10M	-4
ZAIRES	M9	+14
US	N1	-2
LEAPS	5K	-15
BIG	D4	-6
(Q)		-20 = -15

Table 8-10 Another Tremendous Shot

The variation that refutes FLAW illustrates a profound principle; when one player's move is forced, the opponent may make a setup. The point is that the opponent cannot both block the setup and make his forced play. The plays QUA (6H) and QUAG (10K) are obvious follow-ups to FLAW. LEAP blocks one of these and creates a devastating out-threat. The threat of SWUNG (O5) is so strong that the first player has to abandon his plan to play the Q.

The most natural play is AQUA (10J, 14). It was played in the game. The game continuation is in Table 8-11, below, in the leftmost column. The column shows that after Dixon's AQUA, Frank played PUGS (6H, 27, EN) to capitalize on the –S hook of VEXIL, and to deny the same to Dixon. Frank was in bad time trouble in the game. If Dixon had found WAT, for example, even though it is objectively inferior to AQUA, Frank might have been unable to physically complete his moves before his flag fell. In the game, Frank personified concentration; the clock did not bother him a bit. Judging from his comments after the game, he clearly expected the variation that was played in the game, saving EN so he could refute WAG with EN (8N). For example, he quoted the move PEGS (6H), which keeps UN to play at 1N, but does not defeat WAG. Amazing composure under pressure, and amazing psychological mastery, too, since PUGS is objectively an error and Frank only won because he foresaw exactly how Dixon would respond.



Position 8-8 Repeat Position

Dixon basically walked into a knockout punch. If WAG were not played, then Frank's EN would score little. The column labeled "Refute PUGS" shows how Dixon could have won the game by playing AW at 1N. This scores 17 versus WAG's 18, and keeps a G on the rack that will cost 4 points after Frank plays out. The net reduction in scoring for Dixon is 5 points. But without the opportunity to hook WAG, and without having EN (1N, 8), Frank can only score 5 points for his EN, instead of the 14 he scores after WAG. The net gain to Dixon is 4 points, which changes a close loss into a close win.

The best-play sequence after AQUA is in the column of Table 8-11 labeled "Refute AQUA." Frank should take the –S hook of VEXIL, of course, with GUNSEL (6H, 25, P). Frank's P has two out plays, PUP (7E, 11) and OP (J13, 10), so Dixon must minimize her loss by playing WAFT (1E, 10).

Player	Game	Spot	Score	Refute PUGS	Spot	Score	Refute AQUA	Spot	Score
Dixon	AQUA	10J	+14		10J	+14	AQUA	10J	+14
Frank	PUGS	6H	-27		6H	-27	GUNSEL	6H	-25
Dixon	WAG	N5	+18	AW	1N	+17 (!)	WAFT	1E	+10
Frank	EN	8N	-14	EN	9I	-5	PUP	7E	-11
	(F)		-8 = -17	(FG)		-12 = -13	(G)		-4 = -16

Table 8-11 Natural Variations, But Many Errors

Another natural play is QUA (10K). Jeremiah Mead, a top-10 player who resides in New England, suggested QUA during the post-mortem analysis. Edley refuted QUA as in the left-hand column of Table 8-12, and MAVEN's slightly better refutation is in the right-hand column.

Edley	Spot	Score
QUA	10K	+11
ENGS	6H	-25
WAG	J9	+23
UP	1N	-14
(AF)		-10 = -15

MAVEN	Spot	Score
QUA	10K	+11
DEEP	9F	-12
WAG	N5	+18
GUNS	6H	-23
(AF)		-10 = -16

Table 8-12 QUA is Possible, But Beatable

FLAG succeeds because it creates a hook for QUA. There is another way to do this, but it fails, as in Table 8-13. WAG creates an A-hook in the O-column because of AWA. Unfortunately, WAG creates a play for the EGN tile group, which played poorly before. The combination of ENG and UPS refutes WAG.

Nice Try	Spot	Score
WAG	N5	+18
ENG	O4	-15
AQUA	10J	+14
UPS	6I	-27
(F)		-8 = -18

Table 8-13 Another Setup for QUA

We are finished analyzing this complicated position. Let us ponder MAVEN's performance. On downside, MAVEN misses the idea of keeping EGSU to refute QUA (B1) after WAT. For this reason, MAVEN never refutes WAT. Though WAT is likely to work in practice because the defense is tricky, it is objectively a 5-point error. On the upside, MAVEN found the rest of the best moves. Significantly, MAVEN would play these same moves in over-the-board games, regardless of time pressure.

The author has to decide whether to design methods that can defend against WAT. The design decision must weigh how often the situation arises and how often MAVEN misses the best play when it arises.

8.10 Opportunities

MAVEN does not include features that have become *de rigueur* in game tree searching. Perhaps the methods described below would be helpful.

8.10.1 Transposition Table

A transposition table is an associative memory that stores previously searched positions so that the search engine can reuse the search result if the position should occur again [41]. Transposition tables have been intensively studied in the context of chess programs.

MAVEN does not have a transposition table. This was not an oversight; the author implemented a transposition table, but pulled it out after encountering memory-management difficulties. *That* was an oversight.

It turns out that transpositions are unlikely to occur except in one-tile endgames. The reason is that transpositions cannot occur before the third ply, and most endgames are of the out-in-two variety, so there is no third ply. However, in one-tile endgames it really pays to have a transposition table, since the same moves reoccur in different orders. However, it is a historical quirk that the word QAT entered the lexicon at the same time that MAVEN was having these problems, and QAT has the side effect of greatly reducing the frequency of stuck-tile endgames, since the Q can be played much more easily when

QAT is legal. Besides, MAVEN plays one-tile endgames with great accuracy anyway. Nevertheless, if the goal is perfection then there should be a transposition table.

8.10.2 Killer Moves

It is potentially possible to speed up endgame search by reusing moves that were calculated as good in other branches. Such moves are called "killers." Specifically, when following B*'s Disprove-Rest strategy we can speculatively insert a "killer" move into the tree. If the killer does not cause a cutoff then we must generate, but the only goal in many endgames is to narrow a search bound, and this trick can do it. This is particularly true for one-tile endgames. This method is known as the "killer heuristic" [41].

8.10.3 Scalability

The efficiency of MAVEN's approach is clearly a strength, but has been achieved at the cost of accepting a non-zero error rate. Classical full-width search algorithms will play any situation perfectly if given enough time, but MAVEN's algorithm will not. It would be an improvement to use additional time to reduce the error rate somehow. To the best of the author's knowledge, no scalable variant of B* has been published. Possibly a variation such as Palay's PB* [43], which is inherently tolerant of errors would be effectively scalable in a Scrabble application. Error tolerance appears to imply scalability in the Scrabble domain, because the depth of the domain is limited.

8.11 Assessment

The examples of this chapter have shown the difficulties and complexities of endgames, and a great many positions involve MAVEN making an error. The author hopes to convey the complexity of endgames, but if the reader concludes that there is a lot of room for improvement then I have misled you.

To be honest, the author hopes that these examples of MAVEN's errors have convinced you that MAVEN's endgame player is hard to fool. The position has to be complicated before MAVEN chooses the wrong move, and then the move is likely to work out for practical reasons.

Chapter 9 – Pre-Endgame Play

The pre-endgame consists of the moves between the midgame and endgame, when the bag is relatively empty. In terms of strategic decision-making, the key issue is whether volatility is an important consideration. Volatility is the Scrabble concept that corresponds to statistical variance. In practical terms, a volatile position is one in which come-from-behind wins are possible. During the midgame, we have the property that winning percentage and point differential are almost perfectly correlated. That changes during the pre-endgame, because volatility is not fairly constant in that phase.

The first section discusses the new aspects of the pre-endgame. The focus is on how the space of positions changes. The next section illustrates the issues by giving three examples. The architecture MAVEN uses to address pre-endgame requirements is in the third section. The last section is an extended discussion of the pros and cons of the implementation.

9.1 Requirements

A typical pre-endgame consideration is whether one side will come from behind with a bingo. The historical development of MAVEN's pre-endgame analyzer was that some humans maintained that they could beat any computer by keeping things close and then fishing a bingo out of the bag at the end, since computers would not take appropriate defensive measures. It is the author's contention that such assertions were exaggerated extrapolations from small samples. At any rate, the author has witnessed many tournament and casual games between MAVEN and top human experts, and has rarely seen the successful application of any such strategy.

Nevertheless, the potential existed for this strategy to swing 5% of all games. With so much at stake, MAVEN needed to take preemptive action.

9.1.1 Definition of Pre-Endgame

The first question is how to identify pre-endgame positions. The definition of the pre-endgame in terms of volatility is too vague to implement. The simplest definition is that every position after a certain point in the game (which we will measure according to the number of unseen tiles) is a pre-endgame. This heuristic is good because the standard deviation of the future (which we have little influence over) increases monotonically with the number of unseen tiles, whereas the standard deviation of the next two turns (which we may be able to influence) is relatively constant at about 30 points. Therefore, if the number of unseen tiles is large, then the effect of the next turn is small. Consider the manner in which the "endgame-ness" of a pre-endgame position decreases as the number of unseen tiles increases.

When one tile remains in the bag then the true endgame will commence with the next move (barring a pass). Such a situation is a "PEG-1" (Pre-Endgame-1). There are up to eight possible endgames to consider. In theory, we could enumerate all possible one-tile draws from the bag and solve the resulting endgames, which would result in perfect play (provided that the best move is considered). In a PEG-2, there are 36 endgames, and there is a new possibility: the side-to-move can play exactly one tile, which would bring up a PEG-1. We can no longer always reduce the problem to repeated use of the endgame

analyzer, but any move that empties the bag could be exactly analyzed that way. In a PEG-8, it is no longer possible to empty the bag, so no endgames can arise immediately. Every play leads to another PEG. Moreover, there is a plethora of possible continuations, so the impact of the draw is immense.

It is clear from the sequence given above that the value of precise analysis decreases as the number of tiles in the bag increases. In other words, MAVEN's statistical evaluation model becomes progressively more accurate as the game lengthens. We only have to worry about specifics close to the endgame.

A reasonable case can be made for starting the pre-endgame at anything from PEG-6 through PEG-14. Steven Gordon [49] defined the pre-endgame to start at PEG-6 because in his view the key issue is that exchanging is illegal. If you prefer PEG-7, then you regard the ability to create an endgame in one turn to be the defining characteristic of a pre-endgame. Starting at PEG-14 is justified if you regard the possibility of having an endgame on your next turn (i.e., after an exchange of bingos) to be the defining characteristic.

We made a design decision that the pre-endgame begins with PEG-9. This choice is motivated by my belief that most pre-endgames feature relatively low bingo chances, so the PEG-14 definition seems to be overdoing it. Then there is an implementation consideration: we want to enumerate all moves using the unseen tiles, a task that is progressively more difficult as the tile counts increase. Finally, MAVEN specifically uses PEG-9 rather than PEG-8 or PEG-10 because there are exactly 16 unseen tiles in a PEG-9, and that is convenient because information about the unseen tiles can be coded as a 16-bit unsigned integer. (Really deep strategic insight there.)

9.1.2 Pre-Endgame Tactics

Of course, scoring points is still important, as is keeping good tiles. These things are always important, but in the pre-endgame, these factors should be balanced against other issues. Below we discuss some of the issues.

You want to obtain a good endgame. Mostly this means that you want to move first against an opponent who holds an almost full rack. If you move first then chances are you can play out in two moves, which will gain in two ways. You will make one move more than the opponent will, and you will gain from his remaining tiles. This advantage is worth about 25 points on average.

Avoiding the Q can be an issue. If a Q is still unseen then you need to be cautious about drawing tiles. This is true even if you hold a U, since late in the game there is no guarantee that the board will be receptive to a Q.

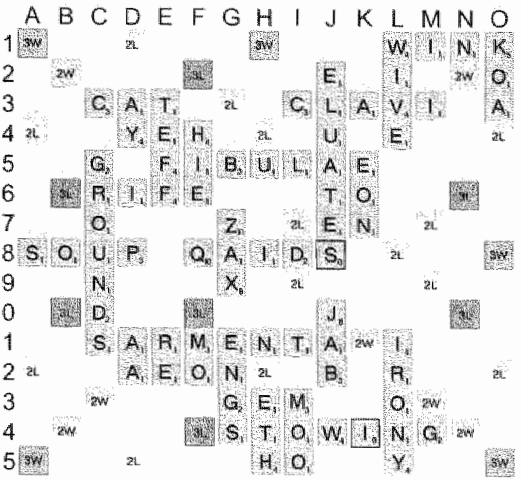
These issues are "timing" issues with respect to the end of the game. You want to score well and to time your plays so that the opponent will empty the bag.

There are defensive issues. You want to block openings that your opponent can use to score well. Usually this means blocking bingo lines, but it can be other things as well. For example, if you see a spot where the Q can be played profitably then you should often block.

Finally, we have fishing. You want to create chances for your own big plays, especially bingos.

9.2 Examples

This section shows five quintessential pre-endgame plays. The moves seem ordinary, but there is always more than meets the eye.



MAVEN:	E, P, R, R, T, V, L, 348
Logan's last:	KOA (O1,7) 332
12 Unseen tiles:	ADDEEIILRSTU

Position 9-1 Pre-endgame Block, And Surprise

In Position 9-1, MAVEN had a small lead over Adam Logan in game 8 of their 1998 match. The unseen tiles are bingoish, so MAVEN had to block the key bingo line, which is the A-column. There is also a bingo line on the M-column, but the spot is less flexible, as it requires a seven-letter bingo ending in S, whereas the A-column allows any eight-letter bingo using an S.

MAVEN played VERTS (A4, 12, PR). VERTS left one tile in the bag, which is the ideal endgame timing situation. It is ideal because if Logan does not have a bingo on this turn then he will not be able to fish again. Leaving two tiles would allow Logan to fish again. Furthermore, emptying the bag would make any bingo that Adam played into a bingo out, thereby depriving MAVEN of a turn.

We have seen that VERTS blocks bingos and has ideal timing. In addition, it is a nice fishing play. VERTS keeps a P and R so that PERVERTS (A1, 39) is possible if MAVEN drew an E from the bag. Alas, this is too little to win if Logan bingos—MAVEN would need a little over 45 points to hold on, I reckon—but it would ice the game if Logan did not bingo.

Position 9-2 is a PEG-1 is from game 5 of the 1998 match between MAVEN and Adam Logan. MAVEN had a lead, but faced three bingo threats. Logan might have had bingos through BIZE (e.g., MISENROL), along row 6 to hook KA (e.g., MOANERS; SKA is legal) and down the B-column to hook PAT (e.g., MOANERS again, making SPAT).

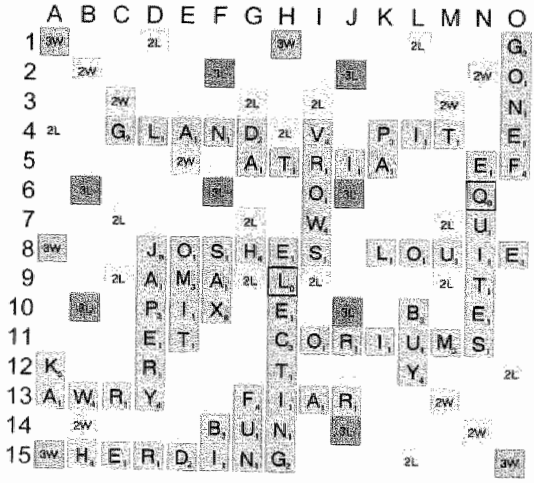
One move guaranteed to win by blocking all threats: OOTIDS (6C, 16, G). Other moves allow bingos, which would narrowly lose. For example, GOOD (3B, 19, IST) is the normal move. GOOD blocks bingos using PAT and BIZE, but allows bingos to KA, so it wins only 7/8 of the time.

MAVEN:	D, G, T, S, O, O, I,	352
Logan's last:	PAT (10C, 13)	293
8 Unseen tiles:	?EMNORSX	

Position 9-2 Exacting Play Required

Positin 9-3 is from a game between Dan Pratt and Ed Halper. It was published in [8]. The author apologizes for not having an example from MAVEN's praxis, but this position illustrates the theme of Q-blocking and Q-avoidance as well as any. Also, the example has the benefit of giving two lessons using one diagram.

In this PEG-9, Dan Pratt sees a Q in the bag, and there are no U's in the bag or useable on the board. Just the previous turn Pratt had exchanged the Q. Halper Halper had played 3 tiles (PIT, 4K, 14), so the Q is in the bag with probability 75%. Pratt has to minimize his chance of drawing the Q.



Pratt:	C, E, E, N, O, S, Z,	280
Halper's last:	PIT (4K,14)	285
16 Unseen:	AADDEEIILLNOOQTV	

Position 9-3 Classic Q Pre-endgame

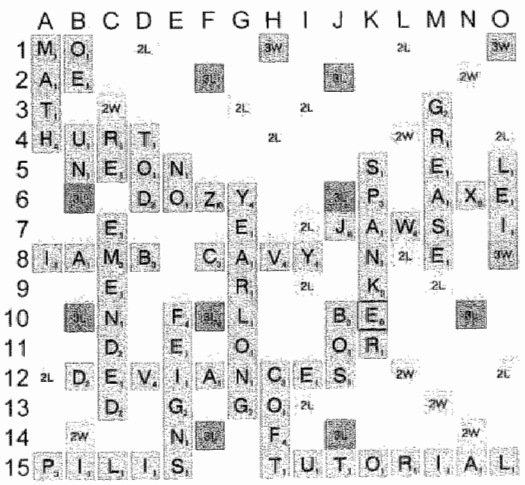
As Joel Wapnick noted [8], Pratt also has to prevent Halper from playing QAID, which is a real possibility considering that the bag contains AADDII, in addition to an available A in GLAND. (Note: this game predated the advent of QAT, so QAID was the only realistic possibility for disposing of the Q.)

Pratt's play was AZO (E4, 24, CEENS). To quote Joel Wapnick, "This play doesn't look impressive, but it is. Not only did is score well while playing away only two tiles, it also blocked both spots for QAID (E4 and 6D)."

One aspect of this position is that AZO, despite its brilliance, might not be the best play. Wapnick recommended RE (J13, 4, CENOSZ), because it keeps the tiles for COZENS (15J, 86). This brilliant shot has an excellent chance to put the game away regardless of the Q. For Halper to hold on, Halper must block the 15th row without drawing the Q. Many players would not realize the threat. Others would see the threat and overreact (e.g., play DOOLIE (15J, 28) to block, but drawing 6 tiles and thus pulling the Q and losing on 6/9 draws). Still others would block, but allow a big Z play (e.g. block with RED (J13, 4) or DO (15J, 7) drawing only one or two tiles but setting up DOZENS (15J, 78)). Others would block properly, but still draw more tiles than Pratt. The only ideal block is a one-tile play like REV (J13, 6), which would truly be a miraculous find. In that event, the exchange of one-tile plays for low scores is a breakeven proposition that keeps AZO in reserve if nothing better comes up.

Wapnick told the author that Dan Pratt felt that RE was weak because Halper might score too well with it. Fascinating! MAVEN's pre-endgame player liked AZO, as played in the game, but simulations (chapter 10) preferred RE. However, it is possible that simulations would not handle Halper's tiles properly in this complicated situation, so I will reserve judgment.

Position 9-4 is an example of bingo fishing. This example is from MAVEN versus Adam Logan, Match Game 12, 1998. The rack seems too heavy to justify fishing, despite the presence of a blank. The board, too, offers little. In addition, making a bingo seems unlikely to help, since MAVEN starts out 73 points down.



MAVEN: H R T T W A 330
Logan's last: LEI (O5,13) 403
9 Unseen tiles: AEIIIOQUU

Position 9-4 Stunning Pre-endgame Fish

However, Scrabble positions often hide deep secrets that simulations reveal. MAVEN noticed that if it played off the W and drew a U then it would have MOUTHPART (1A, 92). This is an amazing find, which was rewarded by a U from the bag and a 2-point victory. By the way, a drawing a U from the bag seems like a long shot, with only 2 U's in 9 tiles. But the chance is higher than it seems, and may be as high as 35%, since the opponent is unlikely to have kept both U's last turn. He might have kept one U to go along with a Q, but if he held two U's then he would have played LIEU (O4, 15) instead of LEI (O5).

Position 9-5 is a PEG-2 from the last round of the 1990 National Championship. Joel Wapnick and I had lunch on the last day, before the awards ceremony. Wapnick was disappointed with his result in this tournament, only 17-10[35]. Characteristically, Wapnick attributed this result to the quality of his play, which he said was not as sharp as it had been a few years ago. Then he brightened up considerably when he produced the diagrammed position.

Wapnick: E, E, E, L, S, T, W, 327
Kramer's last: FRAG (D1, 24) 316
9 Unseen tiles: AAINRRUVX

Position 9-5 1990 Brilliancy Prize

Wapnick explained that when the position occurred he paused to ponder it for some time. Of course, he could play SWEET (O11, 43), but he thought he might lose to VARIX (B2, 62)[36]. If he drew awkward tiles after SWEET, he might even lose to XI or XU (B10, 52), or PAX (B8, 57). He then related that he played STOLE (A5, 5). I strained to disguise my incredulity. Wapnick sacrificed 38 points and an S to block VARIX, keeping EE to boot. It did not add up. Of course Wapnick regained points by blocking—without VARIX and XU there did not seem to be even a 30 point play for the X. Thus, *when the opponent held the X* Wapnick would regain 30 to 35 points. Still... It seemed like too much to give up.

I started to see how it *could* work when Wapnick explained a little more. Wapnick reasoned that his lead would be enough to hold on, since his balanced rack leave virtually guaranteed a good endgame, whereas the opponent might have difficulty going out-in-two. So there is hidden equity: without VARIX, the opponent is forced to play his heavy tiles one at a time, and therefore might be unable to go out in two moves. It followed that STOLE virtually guaranteed that Wapnick would have the last move in the game. That must be worth a few points.

Nevertheless, STOLE cannot possibly regain all the lost points. To post a higher winning percentage despite what appears to be a 14-point sacrifice[37] would require that the opponent's scoring variation be greatly cut down. Wapnick seemed confident that without VARIX and XU the opponent would be unable to put together a big play. I was doubtful, especially since Wapnick had only an 11-point lead. Might Wapnick need SWEET's 43 points?

---

[35]Reflect on this for a moment, as disappointment at a 17-10 result is not something that most of us can relate to. I would be satisfied with a 17-10 score, since such a result (in a Nationals) is a 2000+ performance. A champion's viewpoint differs. If a champion becomes satisfied with 17-10, he will soon find himself out of championship class.

[36]We can see the value of tracking tiles even before the endgame.

[37]My estimate of 10 points is derived as follows: 38 points for score - 30 points for blocking VARIX + 6 points for the rack advantage of EL over EEW = 14 points.

Wapnick submitted his play for consideration for the tournament's brilliancy prize. As luck would have it, I was on the committee judging the submissions. There was some debate within the committee about what standards to use in awarding the prize. Peter Morris submitted one of his moves in which he fished a tile figuring that if he drew an R he could lay down EPICUREAN[38] through two tiles on the board for a 90-point, 9-letter bingo and a come-from-behind win. It turned out that Morris's idea was correct, but he could have fished his tile in another spot for a slightly higher chance of winning. Could we award the prize to Morris, despite missing the best play? One committee member indicated that he would vote for Morris's play, saying, "Peter's play is brilliant. So what if he missed an even more brilliant play." Then I asked how he would vote if Wapnick's move, STOLE, which no one believed could be the best play, turned out to be correct? "If STOLE is best then that's got to be the one." Upon reflection, I agreed with this committee member. I used MAVEN to determine whether STOLE really was the best play in the position.

In this position, all of the plausible moves empty the bag, creating an endgame. My plan was to exhaustively analyze all of the endgames that arise from emptying the bag. By the summer of 1990, MAVEN's endgame player was already well established. Expert players knew of its prowess because of a comparative study I had published and circulated among master players.

Table 9-1 shows how STOLE, SWEET and seven other plays work out. The other seven moves play in the A and B columns to block VARIX or PAX. There are 23 variations, one for each distinct rack the opponent could have held. The 23 racks are not equally likely. Because there are 2 A's and 2 R's among the unseen tiles, racks that use one A (or R) are twice as likely as racks that use zero or two A's (or R's).

Table 9-1 shows the point differentials of the nine moves against each opponent's rack. Any positive number corresponds to a win for the move at the top of the column, and a negative number is a loss. Note that the numbers across each row are highly correlated; in other words, the tiles the opponent draws explain a large amount of the variation in results. Notice that there are substantial variations within a row; in other words, the same rack can work out differently depending on the board.[39]

[38]Zev Kaufman calls Peter's 9-letter game-winning come-from-behind-bingos "Morrisoids." Serving on the brilliancy prize committee taught me that such plays are not accidental. *Peter makes them happen.*

[39]Such juxtapositions of insights allow *true* experts to argue convincingly for *both* sides of an issue.

Rack	Weight	STOLE (A6, 5)	SWEET (O11, 43)	WENT (A1, 18)	WEEST (5A, 25)	WEN (A1, 15)	WEASEL (B1, 22)	WEEP (B5, 13)	WEST (5B, 23)	WALE (B2, 14)
IUNRRVX	1	21	10	10	12	2	7	9	11	9
AUNRRVX	2	16	28	5	5	-2	-4	11	1	-5
AIURRVX	2	11	16	3	13	3	7	-6	2	7
AIUNRVX	4	11	9	16	10	21	9	13	2	9
AINRRVX	2	9	-5	3	-7	-2	-8	11	-5	-6
AIUNRRX	2	24	31	12	13	11	6	16	10	6
AIUNRRV	2	4	76	56	60	50	53	32	55	49
AAURRVX	1	10	28	16	8	39	10	13	-1	-9
AAUNRVX	2	10	28	16	3	41	9	13	20	-12
AANRRVX	1	2	6	-4	-12	-14	3	23	-10	-12
AAUNRRX	1	14	31	8	13	8	10	5	5	1
AAUNRRV	1	11	70	51	52	77	50	43	46	44
AAIURVX	2	23	3	-1	11	33	1	-13	-13	9
AAIRRVX	1	5	9	1	-10	32	-15	-7	-9	-12
AAIURRX	1	16	27	8	11	6	10	-6	8	2
AAIURRV	1	3	44	52	39	46	52	39	39	49
AAIUNVX	1	3	-1	-7	11	-8	-1	-17	-18	9
AAINRVX	2	-4	-1	-4	-13	-4	-17	13	-8	-12
AAIUNRX	2	3	27	7	11	19	10	-13	10	6
AAIUNRV	2	10	51	44	39	43	43	28	26	40
AAINRRX	1	-5	27	3	-5	2	-7	2	-5	0
AAINRRV	1	10	56	40	52	30	29	36	33	33
AAIUNRR	1	9	40	44	46	49	60	35	29	43
	Diff	9.86	24.75	15.78	14.64	20.47	12.33	11.42	9.22	9.92
	Wins	33	31	30	29	28	27	27	25	24.5

Table 9-1 Game Outcome by Move and Opponent's Rack

In my report to the Prize Committee, I gave all variations after all moves. This tremendous amount of information is obviously too much to check by hand, and I doubt that any committee member actually checked all of them. I hope that a diligent committee member verified the variations predicting close victories. Such variations are the ones to check because errors there can turn the game around.[40]

[40]If you decide to investigate yourself, remember that in 1990 we used a slightly different dictionary than we do now. A computer analysis by James Cherry using ACBot disputing some of

The bottom line of the table is, well, the bottom line: STOLE wins in 33 / 36 cases, and SWEET wins in 31 / 36 cases, which proves that Wapnick's judgment was sound. Note that STOLE surpasses SWEET despite a 14-point sacrifice. If you have read Wapnick's book [8], then you know "Wapnick's Rule:" *when in doubt, take the points*. Obviously there must not have been any doubt. Great players know when to break rules.

Do you want to know something eerie? The opponent's rack was AINRRVX, which STOLE defeats, but SWEET loses to. This is how legends are born. ("It was like he could *see my rack...*")

Note that STOLE's edge over SWEET is small. Because SWEET suffers 3 one-point defeats, if Wapnick's lead were even 2 points larger then SWEET would surpass STOLE as the top choice. Similarly, because STOLE often hangs on for 2 or 3 point wins, if the score were even 3 points closer then SWEET would again be best. In other words, STOLE is the best play only when Wapnick leads by 9 to 12 points! This makes Wapnick's insight even more remarkable.

Did you realize how complicated our position is? Given any lead, some move maximizes winning chances from that lead. That move can be computed from the point differentials given in Table 9-1, by calculating how many games are won given each lead, breaking ties by using point differential. The result is Table 9-2.

Wapnick's Lead	Best Move
+13 or more	SWEET
+9 to +12	STOLE
+2 to +8	SWEET
+1	WEEP
-13 to 0	SWEET
-27 to –14	WEN
-61 to –28	SWEET
-62	WEN
-63 or less	SWEET

Table 9-2 Best Move as a Function of Wapnick's Lead

Four moves could be best, depending on the score! What's more, some moves sneak in for just a narrow scoring range; STOLE's range is just 4 points wide, and WEEP would be best if and only if Wapnick led by 1 point![41]

Wapnick's level of insight is truly remarkable.
One has to wonder how he did it. Did Wapnick calculate all these variations? Certainly not! Did Wapnick calculate a few sample variations? If so, then how did he select the variations to try? Perhaps Wapnick just relied on years of accumulated experience in playing tricky endgames.

One insight is that the opponent must hold the X if he is to have any chance, because if Wapnick plays WAX (B2, 26) or EX (D14, 20) then the game is too one-sided for the opponent to come back. You can verify this by noting that the opponent cannot get many

---

these variations apparently did not have the correct vocabulary. I am open to the possibility that there are errors in the analysis, but I will never recheck the conclusions owing to the difficulty of recreating the vocabulary.

[41] In retrospect, this analysis is unconvincing if Wapnick trails by a lot because it ignores fishing moves. It also considers only moves that interfere with VARIX and PAX. Can Wapnick win more often if he trails by 60+ points by fishing a bingo or other crusher out of the bag? A programmer's work is never done...

points for blocking WAX, and that the opponent cannot get many points for playing off the V.

The key insight, which is far from obvious, is that STOLE wins if the opponent holds a U. The opponent must go out-in-two while scoring well with his X, and after STOLE there are no good scores using X and U. It follows that the X and U must be played in separate turns. Thus, the U must go in a move playing the V or RR which also turns 5 tiles, since all the big X plays use 2 tiles. There are no such plays, which means the opponent will always have tiles left. Then Wapnick will win, as shown in Table 9-3.

Points	What for?
11	Wapnick's lead
5	STOLE
-27	For the X play
22	For our W play
-14	Opponent's next play
+7	Our out play
+2	His tiles
= 6	**In total, a narrow win**

Table 9-3 How Wapnick Wins if Opponent Holds U

Therefore, the key bag contents are AU, IU, NU, RU, or UV. Adding these to EEW gives a sick rack, but it should go out in two moves. The plan is to play the U and W in separate turns, since these tiles do not go well together. The W will go with an E and perhaps the other tile from the bag, and the U will go with an E and perhaps the other tile from the bag. Wapnick has at least 15 for EW because of WE (13K), which sets up a nice hook: EAU (M13, 18). The opponent cannot prevent this, because Wapnick has WANE (A1, 21) and LEU (L4, 7) to go out in two. Thus, we already have our variation when AU is in the bag.

Another spot for the W is at A1, where Wapnick can play WINE (21) if an I is in the bag. Can the opponent prevent this using AANRRVX? VARA (B2, 14) is possible, as is RAX (B2, 14). The trouble with RAX is that our WEN (A1) is still possible, so after Wapnick clears vowels with LIEU (K4, 8) the opponent will lose because Wapnick scores 17 for WEN (A1) and 15 for WE (13K): 16 - RAX + LIEU – NANA (A1, 24) + WE + tiles (RV) = 5. VARA fails because LIEU sets up a 15-point play for the WE: WEEN (6J). It follows that the best the opponent has after VARA is AXE (F10, 27), but this loses: 16 - VARA + LIEU - AXE + WE + tiles (NR) = 2.

If NU is in the bag, then Wapnick should do well because of WEN (13J, 22): 16 - VINA + EAU (11G, 6) – OX (J9, 25) + WEN + tiles (ARR) = 4 points. In this variation, EAU (11G, 6) blocks AXE (F10, 27); the variation is a 1-point win if Wapnick plays WEN first, then LEU (L4, 7) to go out. The opponent will not block WEN (13J) because Wapnick has WEN (A12, 19) as a

backup play. MAVEN produced an even finer play: INURE (2A, 18), using the I placed by VINA. This gem of a play leads to a 1-point better spread than my variation.

Now we come to the losing variations. First is AAINRVX, where Wapnick loses because the opponent has a forced out-in-two. After XENIA (F11) the opponent has VARA (B2) and VARA (15F, 21). I think most players would miss this chance, though. To find XENIA would take x-ray vision. Wapnick would do well if XENIA were missed, since when Wapnick goes out-in-two the variations are narrowly in our favor, as before.

The second loser is AAINRRX versus EEUVW. This variation would win in the current tournament dictionary because LUV (K4) helps us go out-in-two. However, in 1990 this lost because the opponent cannot manage an out-in-two.

How much of this calculation Wapnick managed over the board is hard to say. It would take several minutes of careful thought, even for a master of Wapnick's caliber, to produce this type of analysis.

9.3 Architecture

In this subsection, we provide solutions for two of the three issues mentioned in section 9.2. Good endgame timing is resolved by a simple table-lookup indexed by the number of tiles left in the bag after the play. There are several cases, depending on whether blanks and the Q are still in play, but that is just a detail. It is easy to code and creates better endgame timing. This feature emphasizes endgame play, which is to MAVEN's advantage.

Blocking is much harder, and thereafter comes fishing for bingos, which is even harder. Blocking requires, in theory, a two-ply search, and fishing for bingos requires a three-ply search (or simulation). It seems clear that the two-ply problem must come before the three-ply problem (i.e., before fishing), so we focus below on how to block the opponent.

The evaluation function of a two-ply search takes the form of

$$\text{Our Score} - \text{Opponent Score} + \text{Future},$$

where the rack evaluator and endgame timing factors determine *Future*, and the scores are determined by the moves. *Our Score* is known from the move we are generating, but *Opponent Score* must be estimated because the opponent's rack is unknown.

At first, we considered using simulations (Chapter 10), but when we faced the problem (circa 1991), we did not have computers that were fast to simulate plays in real time. Besides, the problem was to *generate* good moves and simulations depend on the quality of the moves fed into them.

So, it was decided to compute an approximate distribution function of the opponent's move. Here is how it worked: we threw all of the unseen tiles into one large "rack" and generated all of the moves that could be played. For each move, we computed the probability of holding those tiles, and created a list of the high-scoring spots and calculated the chances of playing there. Then we noticed that scores in a spot could be different when a move used a Q, in particular, compared with the same spot not using the Q. Therefore, the definition of "spot" was expanded to include a specification of the

highly valued tiles used by the move. (To our chagrin, such obvious facts had escaped our attention during domain analysis.)

One of the key issues is to account for the quality of the tiles used by the opponent's move. If an opponent scores 24 points using a V and W then he has obtained a good result. If he scores 32 using AERSTX then he has gotten a poor result. MAVEN adjusts the value of an opponent's play by subtracting the Basic-1 values of the tiles used. This would adjust a 24-point play using VW to 24 + 4 + 6.5 = 34.5, and adjust a 32-point AERSTX play to 32 – 0.5 – 4 – 1 – 7.5 + 1 – 3.5 = 16.5. The point is there is no point trying to stop AERSTX from scoring—those tiles will score well somewhere—but VW should be reduced to its normal pathetic standing.

Once a distribution is created, we can approximate a two-ply search. For every move of ours, we scan the opponent's distribution to determine which replies are not blocked. Then we weight each reply according to its likelihood.

9.4 Discussion

The pre-endgame engine required extensive debugging. Table 9-4 cites some of the issues that MAVEN has overcome. The pre-endgame analyzer was not reasonably debugged until the simulation player was available as a comparison. It is not thoroughly debugged to this day. The following quote from James Cherry (author of ACBOT) [60] sums up one point of view:

> "ACBOT knows absolutely zero about late-game strategy, and MAVEN's late-game analyzer tries hard, but falls pretty far short of being intelligent, in my opinion."

The author confesses that MAVEN's pre-endgame evaluator has had many bugs. This is particularly true of the versions of MAVEN that Cherry would have evaluated. Curiously, versions of MAVEN that Hasbro distributes are more reliable, but tournament players, like Cherry, dislike using those because of quibbles over the contents of the dictionary. Consequently, these players have not seen how this component of MAVEN has improved over the years.

Issue	Typical Example
Constraints	Project opp will always play the Q, though he is only 7/8 to hold Q.
Tile valuation	If there are two V's in the bag, and a move plays one, to what extent is that valued as splitting a duplicate?
Run-time	With two blanks unseen, the search could run for several minutes.
Tactics	Needed to add a post-processor that checks for setups off our moves.
Tuning	Many parameters, some for rare cases for which we had no experience.
Double count	If opp bingos out, we cannot score the tiles on our rack using Basic.
No count	If opp has so many moves in one spot that we block all of the moves we have saved, we must still project that he will move *somewhere*.
Responsibility	No feature was responsible for penalizing emptying the bag.

Table 9-4 Sample of Bugs in the Pre-endgame Analyzer

MAVEN's pre-endgame evaluator selects the best move on just under 50% of all moves, which is close to the same level of skill as MAVEN's early game evaluator. On the downside, the consequences of a misstep in the late game are large, so the number of points lost is still high.

The engine unquestionably reduces the number of bingos and other big plays by the opponent. However, it makes plenty of weird moves too. For instance, it could defend against threats that are insignificant or sometimes block spots that were favorable on balance. Thus, the engine is sensitive to defensive considerations, but not sensitive to other considerations. Benchmarking the new generator shows uneven results. Yes, the insight is sharp, the algorithm is clever and the implementation is good, but these do change which side wins the game (at least not in MAVEN self-play trials). The explanation must be that the new generator helps when MAVEN is ahead, since it unquestionably reduces the number of bingos played by the opponent. It must hurt when MAVEN is behind, since the overall results are even. What should we do?

The obvious answer is to use the new generator when ahead and the original when behind, but we never did that. We intended to extend the new generator to cover the come-from-behind case. Unfortunately, we never succeeded in doing so. The reason is that it is not easy to extend the framework. The core concept of this technique is to evaluate moves on the basis of *Our Score minus the Opponent's Score*, whereas coming from behind requires including our next turn as well. The framework covers our future turns by using rack evaluation, but this is insufficient for guiding a comeback strategy.

In summary, we regard the pre-endgame generator as a small positive asset to the program. It eliminated one possible weakness in MAVEN (that an opponent might win by fishing for bingos at the end), while it maintained MAVEN's overall level of play. Since humans will be in the trailing situation in most pre-endgames against MAVEN, the defensive orientation is clearly beneficial.

The pre-endgame engine really shines as a generator of moves for simulations, since it finds defensive moves that would otherwise be missed. For that reason alone, the pre-endgame analyzer is worth all of the trouble it caused.

Chapter 10 – Simulation, Scrabble's Unified Field Theory

Simulation leverages a fast but possibly inaccurate evaluation function into a slow but accurate evaluation function. The idea is to play out the game for a few turns using the basic engine to select moves. Moves that have a high average point differential in the simulated games are best.

Before we describe the technical implementation, it is worth answer a few motivational questions. For example, why settle for a slow but accurate evaluation function? Why not implement a fast and accurate evaluation function? The answer is in the first section.

Another question: why does simulation work? There are several reasons why it would not work, but it *does* work. That is the subject of the second section.

Simulation has an interesting historical development. A player suggested the idea to the author after he had carried out the technique *by himself*. Simulation went on to illuminate a great many mysteries of the game. Truly, the positional theory of Scrabble would be immature without simulations. That story is in the third section.

Simulations were used as an investigative tool for almost a decade before MAVEN applied them to selecting moves over the board. MAVEN was not the first to do this, but probably was the first to have a thorough implementation that hit enough fine points to clearly surpass what can be achieved without simulations. The technical implementation actually used in the 1998 match against Adam Logan is the subject of the fourth through sixth sections.

10.1 General Simulation Algorithm

The process of simulation is simple, as outlined in the following pseudo-code:

1) Generate plausible moves.
2) Decide how many moves to look ahead.
3) Select a sample of racks to give the opponent.
4) While time remains for further simulation
    a. Select a rack for the opponent.
    b. For all plausible moves
        i. Follow up that move with the number of turns chosen in step 2.
        ii. At the end, sum the scores of the moves along that variation.
        iii. Fold in an assessment of the racks left at the end.
        iv. Average with all previous results for this plausible move.
5) The move with the highest average is the best.

This process allows for many variations. You can determine which moves should be simulated. You can control the length of the simulated variations. You can evaluate endpoints using any metric (winning percentage and point differential are the obvious choices). You can vary the process that selects moves within the variations of step 4bi. You can modify the condition that stops the simulation process. All of these choices

affect the quality of the decisions that are reached by simulation. This chapter describes an implementation of simulation in MAVEN, but the reader should bear in mind that it is but one of an infinite family of alternatives.

The technique that MAVEN calls simulation has an established practice in other disciples. Simulation goes by the names "rollout", "Monte Carlo search", and the generic name "stochastic lookahead" [54]. The technique has been applied to the games backgammon [55], bridge [56], and poker [57], and in the fields of operations research and optimal control.

These ideas have been floating around for a while. Practical applications have developed within the last 15 years, as computer power has improved to the point where it is practical to apply them to real domains. The threshold for application varies. For instance, rollouts have been used for exploring backgammon since the late 1980's, but to choose backgammon moves in real time using rollouts is perhaps still impractical. Simulations were first applied to Scrabble a couple of years after backgammon, but the threshold for real-time practical application was passed in 1996, or perhaps earlier. The threshold for applicability depends on many factors. In backgammon, for instance, a key issue is that an exhaustive three-ply search provides rollouts with stiff competition. Another factor in backgammon is that players are expected to move quickly. In Scrabble, there is really no other deep search technique, and tournament time controls offer plenty of time for thought.

10.2 Extending the Basic Evaluation is Hopeless

A property that limits the effectiveness of evaluation functions is that they take a "deconstructive" approach to positional analysis. Breaking a position down into independent factors is helpful in that we can trade off factors, but it is self-limiting because a "synthesis" of the position never occurs.

MAVEN's Basic evaluation function incorporates only "first-order" factors. In other words, we do not try to understand any interactions between factors. For example, a W is usually a bad tile, but *suppose that it happens to play well* on a specific board. MAVEN does not identify that situation, and can make an error. That specific combination of events will rarely happen, and when it does it is not a big deal.

It is possible, in theory, to incorporate sophisticated factors into a positional evaluation. One could, for example, analyze whether there are any productive hotspots for a W, and if there are then increase the evaluation weight of the W. However, that would address only that single, relatively miniscule, shortcoming. What about the numerous other potential dependencies? Even the number of *categories* of possible dependencies is huge. Extending the evaluation function is a bottomless pit.

10.3 Simulation as Oracle

One view of the problem is that the evaluation function applies "average" values of individual factors to a specific position. Of what relevance are average values to this position? Usually the relevance is good, but sometimes not.

Let us recall the discussion on how MAVEN's rack evaluation parameters were derived. The value of a tile was the average amount by which it increased scoring over the rest of

the game. Would better positional play result if we had an oracle that could tell us the true values of all evaluation parameters *starting from the current position*? Simulation is the oracle we seek. Simulation provides a whole-position evaluation of moves that have been evaluated within the context of the current position. The mechanism is to average the scores resulting from playing out the game for a few moves. As long as the Basic evaluator is reasonably accurate, the simulation will uncover whatever "tactical" factors happen to exist in the position.

That is why simulation works. In effect, simulation constructs an evaluation function in real time.

10.4 A Pre-Emptive Response to Skepticism

To understand this section you need only understand that simulation consists of using a naïve evaluation function to select the moves for a statistical sample of variations from the search space. This section will state several reasons why simulations might not work, and conclude with an assessment of why simulation nevertheless does work in Scrabble.

10.4.1 Quality of Simulated Moves

Simulation is subject to doubt along the lines of "garbage in, garbage out." For example, James Cherry, the author of ACBOT, has this to say [60]:

> "Simulation, it appears, depends on how good the computer's built-in opinion of what the "opponent's best response" and "our best response" are."

For an extreme situation, consider the impact of playing *random* moves during the simulation. That may be better than nothing, but the Basic evaluator would play better because it will have a higher signal to noise ratio, even though some of its errors may be systematic.

For a less extreme position, consider MAVEN playing against a weaker opponent. During simulations, MAVEN will choose moves as if the opponent is strong. Does this mismatch with reality cause errors?

10.4.2 Statistical Noise

Another issue is the statistical noise of playing out random variations. Steven Gordon published [28] a study of simulations in which a simple rack evaluator was better than all variants of simulation tested by the author. The cause was a small number of iterations, so that noise overwhelmed the differences between the moves. How many iterations do you need before results are reliable?

James Cherry published [60] the data in Table 10-1, showing the results of four ACBOT simulations of the opening rack CCDDEEI, each with 2000 iterations. The data show that three moves finished first in the simulations, and the values of some moves varied by as much as 1.3 points from iteration to iteration. Cherry wrote, "Even 2000 iterations can be not enough to give a conclusive answer. Longer simulations do not always seem to help sort things out much, which is even more distressing. Too bad I'm not doing my thesis on Scrabble, or I would spend the time to chase down exactly why simulation doesn't seem to converge very well." At least he has his priorities in order...

Word	Spot	Score	Sim1	Sim2	Sim3	Sim4
DEICED	8G	24	Best	0.7	Best	0.0
DEICED	8D	24	0.2	0.4	0.3	0.1
DICED	8D	22	0.3	0.1	0.7	Best
DICED	8H	22	0.3	Best	0.6	0.3
DECIDE	8H	24	1.0	1.6	0.4	0.8
DECIDE	8D	24	2.6	3.2	2.3	2.4
CEDED	8D	24	3.3	2.6	3.8	2.2
DICE	8E	12	4.1	4.4	4.6	4.0
DICE	8G	12	4.4	4.5	4.9	3.9
DEICE	8D	20	4.2	4.8	5.4	4.1
ICED	8E	12	4.0	4.7	5.2	4.8
ICED	8G	12	4.7	4.8	5.6	4.8

Table 10-1 Unstable? Opening Rack CCDDEEI

Let us be practical about the matter. It appears that any of the top four moves are all reasonable. Moreover, DEICED (8G) finished first or tied for first on 3 out of 4 trials. If this is the worst we have to worry about then the problem is too easy!

10.4.3 How Many Moves?

Another issue is how many moves you must look at before finding one better than the move preferred by the static evaluator. If the static evaluator is good, then what is the point of simulation? If the static evaluator is bad, then how many moves will you need to simulate before finding the best one?

10.4.4 Which Racks?

Finally, there is the question of which racks to look at. Suppose that your opponent's last play was to exchange two tiles. If you then simulate a rack containing ABCDEFG, are you making a mistake? Is it even possible for the opponent to hold those tiles? That rack does not have five tiles that the opponent would have kept. James Cherry wrote [60],

> "This brings up the following point: we really need a program that makes sensible *inferences* from opponent's plays at *all* stages of the game. Even more basic than that would be a program that takes the score into account when choosing a play. This is crucial both in mid-game, and perhaps even more in late-game, situations."

Is it possible that changes in the distribution of the opponent's rack would change the ranking of the moves? If so, then of what good is the simulation?

10.4.5 Applicability to Scrabble

The doubts expressed in the previous section are important issues that may prevent simulation from being effective in other domains, but in Scrabble, they do not impede success.

The first doubt concerned the quality of the moves generated, on the theory that if garbage goes in, then garbage comes out. Really, that is overreacting. The Basic evaluator chooses the best move 57% of the time. That is perhaps more skillful than any person. Therefore, simulation should provide a good approximation to ideal play. Compare the situation of chess, for example, where a one-ply search represents an awful level of play, yet deeper search nevertheless can leverage that small amount of skill.

The other doubt is about whether MAVEN plays *too well*. Is MAVEN's strong play an unrealistic representation of the actual play of opponents? Well yes, opponents will not play that strongly, but the mismatch does not automatically invalidate the simulation model. The situation is analogous to deterministic full-width search in games such as chess; the search assumes that the opponent will play as strongly as its search engine. It is an invalid model, but one that has demonstrated great practical success.

A key reason why simulation works is that the errors that the Basic evaluator makes tend to cancel. There are two forms of cancellation.

First, a variation consists of an equal number of moves for both sides. This means that the errors tend to cancel along the branches of the variation. If MAVEN plays weakly for one side, then it also plays weakly for the other. This factor makes it less important how strongly the program plays. In the business we cynically refer to this phenomenon as "playing equally badly for both sides."

Second, cancellation of errors tends to occur between the variations that follow two moves. If MAVEN misplays the positions after one move, then it tends to misplay the variations after all moves, since the positions are similar to one another. Thus, if MAVEN's opponent cannot play as strongly as MAVEN's simulated moves then that does not necessarily invalidate the simulation, because all of MAVEN's alternatives would be underrated by approximately the same amount.

Steven Gordon's work [28] hints at the power of cancellation of errors. Gordon performed simulations where the moves of the simulation were selected strictly based on score, without considering rack leave. Gordon observed no difference between the results of such simulations and simulations in which a rack evaluator was used. By comparison, the difference between the Basic evaluator and perfect play is rather small.

Cancellation of errors is an important principle of the design of simulations. For an example from another domain, backgammon engines systematically enumerate the dice rolls for the early variations of their rollouts [38]. In Scrabble, cancellation of errors is enhanced if you use the same sequence of tiles from the bag for all variations of an iteration, randomizing only after exchanges. This procedure maximizes similarities within the variations of each iteration.

The short answer to the question about statistical noise is "900 iterations." The long answer is that an individual trial has a standard deviation of 30 points, a value that is remarkably consistent from move to move and game to game. It follows that 900 iterations would narrow the estimate of the mean to a standard deviation of one point, which is sufficient accuracy for practical purposes.

The example of James Cherry's 2000 iteration simulations is easy to explain. In 2000 iterations, the standard deviation is reduced to about 0.7 points. In Cherry's table, every value is within 0.7 points of its mean. (The deviation we measured of 1.3 points was between the lowest and highest of four measurements, which is approximately twice the standard deviation from the mean, as it should be.)

The key criterion for choosing 900 iterations is that a one-point margin is "good enough for practical purposes." The "practical purpose" is to win the game, which we achieve by playing good moves. If two moves differ by less than 1 point then it is hard to distinguish them as a practical matter. Thus, Cherry's table is not troubling at all. Nor is it troubling that different simulations might prefer different moves. That simply means that the moves are equally good, which is a property of the moves that simulation properly reflects. The convergence of simulation is not slow at all.

The issue of how many moves is a good one, which we will address in a moment, but there is a subtle issue in the comparison. The question, "how any moves must we simulate" begs the question, "for what purpose?" The implication of the question is that nothing less than perfection will do, but that is not the goal at all. The simplest possible goal is to improve upon what you can achieve with static evaluation alone, and to accomplish that you only need to simulate *two* moves. That is the simplest answer to the question.

To raise the standards a little bit, we can get a more complicated answer. If you want to achieve perfection, at least within the limitations imposed by imperfect static evaluation functions, then how many moves must you simulate? The answer is that you can search more moves as computers get faster. However, because the Basic evaluator ranks the best move first on 57% of all turns, there is little need for simulating a zillion moves.

The question of which racks to simulate is interesting. On a theoretical level, it is necessary to exactly match the distribution of unseen tiles to all information available from the history of the game. In practice that is unnecessary because good moves tend to work out regardless of the opponent's tiles. There are extreme cases, but keep in mind that an average move turns over 4.5 tiles, so the opponent could hold pretty much anything. In a different domain (e.g., the game of bridge) then inferences would be huge, but in Scrabble, you rarely need to consider inferences. When you do consider inferences you usually end up playing what you would have played anyway. Moreover, when you change your mind, it can work out badly.

If you harbored any of these doubts, then the author hopes that this section has at least indicated that specific properties of the domain work to counteract the ill effects.

10.5 Historical Development

The start was a fruitful discussion with Ron Tiekert at a tournament in December 1987. Tiekert was (and still is) a legendary player, the 1985 National Champion, and the player who achieved the highest NSA (National Scrabble Association) rating,[42] and is to this day one of the top players in the game.. Tiekert was acclaimed for his skill at positional evaluation, and the author hoped to learn from him. Evaluation is, after all, the only mistake that MAVEN ever makes. The reasoning behind this statement is that for every game an AI program only makes evaluation errors, provided solely that the program generates all legal moves (as MAVEN does).

One historically important position was the opening rack AAADERW. There are two prominent moves: AWARD (8H, 22, AE) and WARED (8D, 26, AA). Expert opinion favored AWARD, and MAVEN agreed. In dissension, Tiekert chose the unusual play AWA (8G, 12), asserting that it was by far the best play. Tiekert said that he was initially attracted to AWA because it had a nice balanced rack leave (ADER) that is more likely to make a bingo on the next turn than the two-vowel leave of AE after AWARD. In addition, AWA does not open access to the double-word squares, whereas AWARD does. AWARD can be "hooked" with an S for big plays down the M-column.

But Tiekert was not certain that AWA was best until he played out both AWA and AWARD using parallel racks *fifty times each*.[43] Tiekert said that AWA finished with much better results, so he was confident that AWA was better. Tiekert mentioned, in passing, that with MAVEN being as fast as it is, it should be able to crunch the calculation pretty quickly. For the author, this experience was like a ray of light from heaven.

Here is Tiekert's procedure. First, he drew racks for the opponent. He ensured that the racks had a balance of tiles representative of the unseen tiles. Then he played a move for the opponent after each candidate play. Then he filled the rack of the side to move, using parallel draws to the greatest extent possible. Then he played a move for that side. He totaled the scores and folded in an estimate of the value of the two racks left at the end.

Word	Spot	Equity	Opp	Our	Opp Bingo	Our Bingo
AWA	8H	27.2	31.0	47.8	16.5%	39.7%
AWA	8G	27.0	34.9	50.8	16.4%	39.0%
AWA	8F	26.8	30.3	46.6	16.4%	37.9%
AWARD	8H	23.1	36.2	38.6	21.6%	22.8%
AWARE	8D	22.4	33.9	28.9	21.6%	24.0%
WADER	8D	19.5	36.9	31.2	22.7%	11.5%
WARED	8D	19.2	37.1	30.9	22.4%	11.7%

Table 10-2 Simulation Results for Opening AAADERW

[42] 2170, a record that still stands.

[43] I wonder if Tiekert was inspired by New York-area backgammon masters, who were at that time uncovering the mysteries of backgammon by using manual rollouts. Tiekert was a habitué of New York's infamous Game Room, where the city's strongest game players gathered in the 1970's. Tiekert would surely have known some of these pioneering backgammon players.

Results from the original simulation do not survive, but a recent version of MAVEN has run this case and produced Table 10-2. The Equity column shows the net points gained by the side to move over the 3-move sequence. The Opp Move column shows the average score of the opponent's reply, and the Our Move column shows the average score of our follow-up play. Opp Bingo and Our Bingo are the bingo percentages of the next two plays.

Note that AWA finishes well on top (the 8H placement slightly preferable), with AWARD second, and AWARE and WARED bringing up the rear. The key differences are in bingo chances. AWA decreases the opponent's chance and increases our own.

AWA varies in value depending on your placement: it does best at 8H, worst at 8F, and intermediate at 8G. Probably 99% of players would guess wrong if asked where AWA would have the worst result. The 8G placement is no worse than the others are. A vowel/DLS square adjacency just is not dangerous.

Simulation provides the deepest insight into Scrabble positions. It is a tremendous tool. You can control the moves to simulate and the number of moves you look ahead; moreover, you can choose whether you evaluate endpoints by score or by winning percentage. Any position can be evaluated using this technique. The only limitation is whether MAVEN is playing the variations well. Even that limitation is lessened by the fact that MAVEN plays "equally badly" after each candidate. Such "cancellation of errors" is an important factor in simulations. The only remaining drawback is that simulations are long. The standard deviation of a two-move sequence in Scrabble is about 30 points. This means that even after 900 iterations you have a 1-point standard error. If you want to achieve fine judgments it can take a simulation that runs overnight. Still, it is at least possible.

10.6 Impact of Simulation on Tournament Scrabble

By the start of the 1990's, MAVEN had proven that the game was not well understood, and it had proven specific theories about rack and positional evaluation. However, these results were known only within a small community of expert players that had worked closely with the author. We started several initiatives to publicize MAVEN, with the goal of increasing MAVEN's credibility within the expert community.

The first initiative was to publish a monograph [45] concerning MAVEN's endgame skill, as described in Chapter 8. That monograph established credibility with a wider, but still small, audience of experts.

The author's second effort was to publish simulation results, primarily through three media. First was the *Scrabble Players News* (SPN). Editor Joe Edley encountered MAVEN at its first tournament, (and departed the board with a rueful countenance after a classic MAVEN romp) and had turned into MAVEN's biggest and earliest promoter. Edley wanted the author to write notes for a Scrabble game to be published in SPN. I carefully simulated all of the moves and wrote clear, simple notes appraising the moves. The same game is in Appendix B with slightly different notes. The annotations were well received, and Edley invited the author back for two other annotations later on. Edley eventually decreed that all games must be validated by computer analysis. Without this basic safeguard, the annotator would probably miss something.

MAVEN's simulation capabilities were packaged in a simple application, initially available for Macintosh computers and later for Windows as well. Users could set up any position and select any interesting moves for analysis. Users could simulate the position for as long as they liked, using any number of moves of lookahead.

Another promoter of MAVEN's capabilities was Nick Ballard, a master Scrabble player who is better known as a Backgammon World Champion. Ballard published a newsletter, dubbed *Medleys*, for three glorious years. Every issue of *Medleys* was jammed with computer-based insights. Many of these involved MAVEN. For example, every issue had a Consensus Game, in which moves were chosen according to the votes of readers, backed up by whatever analysis the readers submitted. Many readers submitted MAVEN simulation results to support their opinions. Ballard organized and published the analysis, and often relied on MAVEN to validate assertions made by players. Ballard also did several large-scale studies using simulations, including the Basic rack evaluation model.

Then there was self-promotion. After *Medleys* folded (as all Scrabble publications other than SPN eventually do), the author published a series of 15 monographs entitled *Rack Your Brain* (RYB). The highlight of each issue was an in-depth analysis of a single position. The analysis typically occupied from 3 to 6 pages, containing text, tables, diagrams, formulas, and whatever else was needed to illuminate the complexities.

Aside from the results published in SPN, *Medleys*, and RYB, there was a growing culture of experts sharing simulation results. A group of four strong experts from Minnesota (Charlie Carroll, Lisa Odom, Jim Kramer and Steve Pellinen) actively used MAVEN for analysis, and published several results. Jim Geary published a newsletter after the demise of RYB in which simulation results played a role. An on-line community started using MAVEN for simulations, with Joel Sherman and Jim Geary taking active roles.

In addition to these public displays of the value of simulations, several experts analyzed all of their games using computers. The practice of checking for missed moves was available to users of TYLER and CROSSWISE, although the latter would only give you a list of moves in decreasing order of score. However, MAVEN simulations took the process of self-criticism to a completely new level.

Many of MAVEN's early adopters have had huge success in tournament play. Did MAVEN's early adopters win so many events because they adopted MAVEN, or did they adopt MAVEN because the type of people who win tournaments would also adopt MAVEN? Good question. Obviously, the biggest reason for their success is that they were great players to begin with. Still, the author likes to imagine that the expert community has benefited from MAVEN.

10.7 Simulation as Investigative Tool

This section will show some examples of the power of simulation. The idea is to use simulations to determine the truth about a position. At a superficial level, the truth about a position is expressible as a point differential. Usually that is as far as the analysis needs to go. Sometimes it is helpful to examine the variations seen during simulation, so that a more dynamic view of the situation emerges.

The LUG versus GROSZ controversy is a good example. I heard about the opening rack ?GLORUZ from Peter Morris, who was engaged in a theoretical dispute over the relative merits of GROSZ (8H, 48, LU), and a variety of dumps/fishes, of which LUG and GUL (8G, 8, ORZ?) stand out. Fishing must be best, because otherwise the author would not write about it. Table 10-3 should surprise no one.

Move	Spot	Net	Opp	Us	Opp Bingo	Our Bingo
GUL	8G	46.6	26.4	49.7	16.0%	30.2%
LUG	8G	45.1	27.3	48.8	16.3%	31.0%
GROSZ	8H	44.1	37.9	35.6	19.1%	18.9%
RUG	8G	43.4	27.8	49.1	17.2%	32.8%
ZORIL	8D	39.5	35.3	30.5	19.1%	11.9%
LOUR	8F	37.2	31.5	41.1	18.8%	18.8%

Table 10-3 Results for Opening Rack ?GLORUZ

GUL and LUG surpass GROSZ because of tremendous combined offensive and defensive superiority. After GUL, the opponent's score (column "Opp") drops 11.5 points, and ours (column "Us") rises 14.1 points. The combined 25.6 point difference is enough to overcome a 40 point sacrifice because we usually keep the blank: we bingo 30.2%, so we keep the blank slightly under 70% of the time.

Of course, GROSZ has the edge when the opponent holds the Q because the U in GUL allows him to get rid of it. Q is the only tile for which GROSZ surpasses GUL—GUL's advantage is consistent. Note this advantage of simulation: one can examine the variations to determine *why* the evaluation is correct. One can isolate such variables as the tiles in the opponent's rack for further analysis. One can determine whether the opponent exploits specific openings. These data help to explain simulation results in terms that humans can understand. A table of values would only go so far towards satisfying the curiosity of a human expert who has been playing a situation incorrectly for years.

Note that GUL is slightly better than LUG because the hook chances favor GUL. When the opponent holds E, he has LUG-E, and when he holds G, he has G-LUG. Predictably, GUL outperforms LUG in these cases. In contrast, when the opponent holds L or F then LUG is clearly superior, because of GUL-L and GUL-F. Then there is the S-LUG or LUG-S versus GUL-S, and P-LUG versus GUL-P, which are equal. So there are hook chances after either move, but the huge chance for LUGE predominates.

To confuse the issue, note that the X plays better against GUL than against LUG. The best score for an X comes from the J6, J10, F6, or F10 triple-letter squares, and GUL's hooks are more convenient for this purpose than LUG's: FIX, FAX, FOX, LAX, LEX, and LOX versus GOX. The LUG-E chance makes little difference because AXLE, NIXIE, AXMEN, etc., are all longer than 3 letters, so the combined frequency is low. Also, note that -AX plays along the 7th row (and -OX plays along the 9th row) score an extra point after GUL: BORAX (7D) is 28 after GUL, but only 27 after LUG. Overall, GUL has a slight edge defensively because hook considerations outweigh the "X-factor," which arises only 7.5% of the time.

One last observation: LOUR fares terribly, despite being philosophically similar to GUL. The reasons:

1) By playing an extra tile, LOUR gives the opponent extra bingo chances.

2) By playing another vowel, LOUR gives the opponent extra overlap chances. While not generally significant, the overlap chances are important here because our rack retains no overlap chances.

3) The rack ?GZ is singularly unharmonious. Our bingo chance drops from 30.2% to 18.8%.

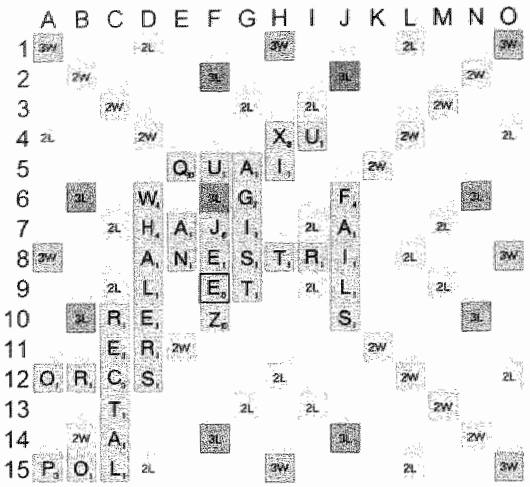
LOUR shows that the correct idea (a dump/fish) is not enough—you must implement the idea accurately.

By the way, many players scoff at GUL and LUG because of a practical consideration. GUL appears to risk 40 points (the score difference between GROSZ and GUL) to gain 2.5 (the difference between the simulated equity). You would therefore require 93.75% confidence in the simulation results to justify GUL. This is a good example of how fear, uncertainty, and doubt paralyze humans. GUL regains a significant amount of equity on rack evaluation grounds alone. The Basic model shows GUL's ?ORZ rack as worth 24.5 –2.5 +1 +3 = 26, while GROSZ's UL rack is worth –4.5 –1.5 = -6. In words: you regain 32 points just on pure rack considerations alone. It is easy to believe that not opening access to double word squares must be worth a few points, too. The degree of faith you need is lower than it seems. Still, few humans would venture GUL instead of GROSZ.

The author believes that GUL is superior to GROSZ, but can imagine that the conclusion is wrong because of a modeling issue. Specifically, the variations examined during simulations are selected on the basis of Score + Rack, whereas in the actual game the moves are selected by simulations. Under TWL98, the average improvement of a move selected by simulation over a move selected by Score + Rack is about 1.5 points. Therefore, in the two-move sequences after GUL and GROSZ there is enough room for error to overcome a simulated advantage of 2.5 points. One could imagine that simulations are less valuable in the variations after GUL, because the move decision tend to be obvious, whereas simulation may have great value after GROSZ. The only way to be certain is to simulate simulated games. This may be a fruitful area for further research.

A panel of experts analyzed Position 10-1 in *Medleys*. Eight moves were rated as best by the members of the panel, and MAVEN's Basic evaluator preferred a ninth move! The consensus analysis talked over the issues from many angles, but just could not agree on anything.

Such disagreement is typical for consensus analysis, by the way. The art of Scrabble analysis is simply not advanced far enough to allow firm conclusions when confusing cases arise. This example will show how expert players can talk themselves into making serious errors, while basing their opinions on concrete observations of the position. Then we will see how simulation views the situation.



Position 10-1 Hidden Positional Issues

The consensus play was AY (D14, 24), though it garnered just 26% of the consensus vote. Supporters and detractors of AY matched each other with points and counterpoints. Table 10-4 shows the key points of disagreement.

Supporters of AY said:	Detractors of AY said:
Block opponent's plays from 14B to 14F.	We need 14B-14F to come from behind
Good POLY- extension plays on row 15[1].	Those are not one-way openings.
CIOPT is good. High bingo chances.	Low turnover. C is bad. Want to keep Y.

Table 10-4 Point and Counterpoint Regarding AY

High-powered players (Kramer, Logan, and Silberg) favored COPRA (B9, 23), supporting their preference with positional reasons. But Rosen, Carroll, and others strong players said that COPRA was poor, and flatly denied the assertions of COPRA's supporters, as in Table 10-5.

Supporters of COPRA said:	Detractors of COPRA said:
COPRA leaves a tight board.	We should not block when we trail.
COPRA keeps Y to score on K-column	The ITY leave is poor.

Table 10-5 Point and Counterpoint Regarding COPRA

CAY was generally preferred in a head-to-head comparison with the similar move PAY. These moves score the same, they occupy the same squares on the board, and they differ in only one tile on the rack. Nevertheless, the panel could not agree on how to rank these moves. Table 10-6 gives the issues.

Supporters of PAY asserted:	Supporters of CAY asserted:
Keeping C is better than keeping P.	P is better because it makes more overlaps.
If you draw an I, bad because it duplicates, then you can play POLITICO (15A, 39)	Opponent will probably block POLITICO before you even get the chance to move.
It is useful to keep new possibilities of play on the L-column.	Hinder opponent replies by placing the C to reduce overlap chances.
What? You need more than that?	The opponent can make SPAY.

Table 10-6 Comparison of PAY and CAY

Can simulations reveal the truth about a position when a dozen experts disagree? An issue of *Rack Your Brain* [68] examined these assertions in elaborate detail. We will condense the analysis to address only the factors mentioned above.

The quality of the CIOPT rack leave after AY was a big source of disagreement. Many players felt it was poor, but this is an "optic" illusion. As the board stands now, the tiles CIOPT would make a bingo on 21% of the two-tile draws from the bag. This is actually somewhat higher than normal, despite the low quality of the tiles. A key factor is the possibility of several prefixes and suffixes: CO-, OPT-, and -IC.

In addition, a hidden factor of CIOPT really sets it apart. There are extensions to POLY (15A) that play along the 15th row that are almost one-way openings, as shown in Table 10-7. These words score well. For example, the lowest score is POLYGLOT, which scores 42. POLYTYPIC, which needs only a Y from the bag, scores 63. The draws S, D, Y, GL, HN, ER, and RU work, for a total chance of about 20%. Particularly important is that a single tile (S, D, or Y) is all that you need. In addition, there is little overlap between the tiles needed for extensions and those needed for bingos.

POLYCOTS
POLYURIC
POLYGLOT
POLYPORE
POLYPHONIC
POLYPOID
POLYTYPIC

Table 10-7 POLY- Plays after AY

Therefore, analysis shows that CIOPT is a great leave here. CIOPT is worth about 7 points, whereas its normal Basic evaluation is –6. This evaluation puts AY in the driver's seat for the top spot. Even the 6 extra points scored by TAPIOCA (14B, 30, Y) are eclipsed by the rack value of CIOPT, and TAPIOCA has other disadvantages—a poor leave and a big triple-word opening.

Another issue discussed by the consensus panel was the 14B-14F opening. Some thought it would be open for our use next turn, for example for a play of POLITICO if we draw an I after PAY. Other players thought we must block it.

Simulation shows that if we do not block then plays in the 14th row make up about 50% of the opponent's moves next turn. In part, this is because good scores are possible there, and in part because this board has few hotspots for non-bingo moves. Blocking is correct by a large margin.

Because the opponent takes the 14th row whenever possible, we can evaluate the new openings created by PAY. Are they favorable or unfavorable? Well, obviously favorable,

if the opponent moves elsewhere more often then not. Without new moves off of PAY, it is hard to see where we can make a play next turn. The disagreement over CAY and PAY shows how humans overdo defensive considerations. Some players *automatically* consider any newly created opening to be a negative. Experts preferred CAY to PAY by a 17 to 7 margin.

Move	Spot	Score	Rack	Opp Next	Our Next	Net Equity
AY	D14	24	CIOPT	29.9	42.1	35.1
COPRA	B9	23	ITY	25.5	32.8	28.9
PAY	K5	26	IOPT	36.9	39.5	25.2
CAY	K5	26	CIOT	33.6	34.5	23.3
PATIO	B6	20	CY	37.7	41.4	21.9
TYPO	C3	23	ACI	36.9	37.2	21.4
TAPIOCA	14B	30	Y	44.1	32.1	17.3

Table 10-8 Simulated Equity of Moves for Position 10-1

Table 10-8 shows simulation results for some of the moves proposed by the consensus panel. The tremendous defensive advantages of COPRA and AY are apparent. AY cuts the opponents' next turn to 29.9. COPRA takes that a step farther by taking away the B-column bingos; COPRA leaves the opponent only a 25.5-point turn. Amazing!

The offensive advantages of AY stand out. AY gets our next score up to 42 points! Those bingos and POLY- extensions really pay off.

A few numbers from the table are expected. MAVEN's PATIO (B6) creates an overlap on the A-column that increases the opponent's score and ours. TYPO creates a hotspot on B6, again raising the opponent's score and our own. TAPIOCA's defensive shortcomings are severe, and they are not compensated by increased offensive chances.

Most of the moves suggested by the panelists lost a ton of points. The move preferred by the Basic engine, PATIO (B6), nets 13.2 points less than AY. The author's move was PAY, which loses 9.9 points. The error analysis is in Table 10-9.

Move	Spot	Score	Experts	Points Lost
AY	D14	24	3	0.0
COPRA	B9	23	3	6.2
PAY	K5	26	3	9.9
CAY	K5	26	3	11.8
PATIO	B6	20	1	13.2
TYPO	C3	23	2	13.7
TAPIOCA	14B	30	1	17.8
ATYPIC	110	18	1	28.9
Average			**17**	**10.0**

Table 10-9 Errors of the Panelists for Position 10-1

Wow! The average panelist dropped 10 points. It makes you wonder how we can claim to be experts. We misevaluated an ordinary position, even given all the moves and all the time for analysis!

Before the advent of simulation, arguments such as this would just go around in circles. Now there is a way to test theories about the game. For example, in this situation we have Cappelletto arguing for TAPIOCA. Without simulation, it would be hard to argue with him. He has been one of the world's best players since he was a teenager, and he won the Nationals in 2000 and the Worlds in 2001. With simulation data to back up what you say,

the positional issue is elevated above the question of which authority said what, and focuses on the question of what was said.

Simulation allows us to study questions about winning percentage. For many years people have made extraordinary claims about how maximizing winning percentage often required moves that sacrificed point differential. With simulations, we can at last compute how often a play wins by simulating to the end of the game.

For example, Mark Watkins, the author of BOBBOT, posted some analysis in response to analysis that Jim Geary had posted to the Internet. The subject is the opening rack BRONZER, where you can place your bingo at 8H to get the Z on the double-letter square at 8L. This scores 106 points, whereas a placement at 8D (putting the B on the double-letter square at 8D) scores 92. A difference of 14 points would normally make the play clear, *but* the 8H placement runs BRONZER up against the triple-word square at O8, so the first player to hold an S is guaranteed a huge score on the O column by pluralizing BRONZER. Jim Geary had used MAVEN simulations to show that the 8H placement averages higher point differential despite the hotspot, with the implication that the 8H placement is the better move. However, Watkins had this to say:

> "On your web page http://www.primenet.com/~jaygee/FALLACY.HTM, you mention the BRONZER rack to start, and consider whether it should be placed at 8D or 8H. You conclude that equity-wise, it should be about 5 pts better to slap it at 8H. My first thought was that win%-wise an 8D positioning might be better. As for any use you make of the spot will just super-increase your lead, while the opponent's use of it allows him/her to mount a comeback. Data from 10000 SOWPODS sims with BOBBOT, infinite depth, with average scores and rack values being noted for the first four ply:[44]

Move	Spot	Score	Opp	You	Opp2	You2	Rack	Diff	Win%
BRONZER	8H	106	61.9	55.8	48.8	46.0	-0.3	97.4	75.2
BRONZER	8D	92	43.0	44.7	45.0	44.3	-0.1	93.2	77.0

Table 10-10 BOBBOT Simulation Data for Opening BRONZER

> As suspected, 8H wins the equity battle, but 8D wins the win% war. After 5 plays, the 8H placement had a lead of 50 or more 72.7% of the time, while the 8D placement had a lead of 50 or more 82.4% of the time. I would be fairly certain that OSPD data would be similar. The only danger in interpretation I see is that the BOT doesn't necessarily shut the board down when ahead, but that should affect each move about the same."

This shows the level of discourse among experts who employ simulations for positional analysis.

This section substantiates the assertion that simulation provides insights that we simply could not obtain any other way. Consider the difference between BRONZER 8D and BRONZER 8H: it is less than 2%! We would have no means of detecting such small

[44] I modified the layout of Watkins's table to fit this page, but the data is his.

differences without simulations. With simulations, we now know for certain that the 8D placement is better.

10.8 Simulation in Real-time – Theory

Having seen the power of simulation, the first thought is that a genuinely awesome player would result from selecting moves in actual games using simulations. True. Unfortunately, the CPU burden of simulations is so high that it is hard to pull it off. Actually doing it well had to wait for several years.

10.8.1 How Many Iterations?

First, we can determine when simulations are beneficial by considering the typical difference between the best and second-best plays. A simulation is trying to distinguish between these plays by a taking a statistical sample. Accordingly, a simulation needs enough samples to differentiate the top two plays.

The data from Table 10-11 are from a 14-game match between Adam Logan and MAVEN sponsored by AAAI-98. MAVEN analyzed every move decision in detail using simulations. The table shows the distribution of point differentials between the best and second-best move for the 309 non-endgame moves of the match.

As you can see, many best moves are almost indistinguishable from the second best move. It would take an impractical number of iterations to ensure that you pick the best move. Yet, the beauty of the situation is that if the moves are so close then it does not matter whether you make an error.

Diff	Cum %	Diff	Cum %
<0.5	12.0	<7	75.1
<1	23.0	<7.5	77.0
<1.5	31.1	<8	79.3
<2	38.5	<10	80.9
<2.5	46.0	<12.5	84.8
<3	50.8	<15	88.3
<3.5	56.6	<20	91.6
<4	59.2	<25	93.5
<4.5	61.8	<30	95.1
<5	68.6	<35	96.8
<5.5	70.6	<40	98.4
<6	71.2	<45	99.7
<6.5	73.8	<55	100

Table 10-11 Distribution of Difference Between Best and Second Moves

To make matters concrete, suppose that you simulate until you can estimate the standard deviation of the difference between the best and second best move down to 1 point. How much error would that contribute?

Any moves that have difference > 3 are three standard deviations apart; so we are guaranteed to select the best move. Moves that have difference > 2 have >95% chance that those moves will be distinguished, and a loss of no more than 3 points if we make an error, so the average loss is less than 0.15 points. 21.5% of moves having a difference between 2 and 3. For moves with a difference between 1 and 2 the chance is less than 16% that an error will be committed, with a cost of up to 2 points, so only 1/3 of a point lost on such situations, which make up 15.5% of all moves. If the difference is less than 1 then the chance of making an error is less than 50%, with no more than 1 point at stake, so only 1/2 point is lost on average. Such situations make up 23% of the total. So add it up: the error rate is less than 21.5% * 0.15 + 15.5% / 3 + 23% / 2 < 0.2 of a point! Since you average 10 non-endgame moves per game, the error rate is less than 2 points per game. Obviously, this is sufficiently

accurate. This is so accurate that the engine will lose more points by omitting key moves than by misevaluating.

To have an engine accurate to that level (i.e., 1 point standard error in a comparison of two moves) would require simulating no more than 1800 iterations, since there is a 30-point standard deviation per trial and 2 * 30 * 30 = 1800. However, because certain safeguards are built into the implementation, in practice we can reduce this number somewhat. For instance, MAVEN's rack samples are repeated for all moves in the simulation, which ensures that every move is measured against the same distribution of racks. Furthermore, the sample is constructed such that each tile occurs at (nearly) the correct frequency, which further reduces the variance. In fact, in MAVEN's implementation the standard deviation of the *difference* between two plays is 30 points, which is half of the variance predicted by *a priori* statistical theory. This accounts for a previous assertion that only 900 iterations are required. MAVEN's implementation for the 1998 match limited the number of iterations to 1000.

10.8.2 How Many Moves?

Table 10-12 shows the position of the best move within the move ordering of the Basic evaluator. This chart omits pre-endgame situations from consideration, leaving 284 moves taken from MAVEN's 1998 match against Adam Logan. The data show that the top move according to the Basic evaluator simulates best in 57% of all turns, and we estimate that simulating the top 10 moves would result in 95.8% of best plays. This would make one error every other game.

Data from John Babina show similar characteristics [69]. His sample shows that SOWPODS simulations using PIOBOTJR selects the first move 63% of the time, and the second move 16%. The 7th, 8th, 9th, and 10th moves in the order each are selected about 1% of the time.

One technical factor is how we made the projection for the "11 or higher" category. Our data (and also Babina's) only included 10 moves. We estimated the "11 or higher" category by counting how often Logan played a move that MAVEN failed to generate and turned out to top the simulation. On the assumption that Logan will find 50% of the moves that MAVEN misses, we can double that number, and we can double that number because we only have Logan's opinion for half of the turns.

Move Rank	Best Play %
1	57.0
2	13.7
3	6.3
4	4.6
5	2.8
6	5.3
7	2.5
8	1.1
9	1.4
10	1.1
>=11 (est)	4.2

Table 10-12 Distribution of Best Move within Basic Ordering

Clearly, that technique is open to doubt. The rate at which the percentage drops through the 7th, 8th, 9th, and 10th positions does not suggest that the residual will be as low as 4.2%. Therefore, we are open to the possibility that the percentage is higher. We could be missing as often as once per game.

This, too, is not as bad as it seems, because the lower the moves go on the Basic evaluator the greater the initial sacrifice of points and rack leave that must be overcome.

The author speculates that the loss in points is greatest when moves are missed near the top of the ranking. Alas, the existing data is insufficient to test this theory.

It seems that simulating about 10 moves should result in good play, but simulating more moves could be better. The author has collected a small amount of data from simulations using 20 plausible moves, and these confirm the estimate of 4%, but this analysis is still preliminary, so I will refrain from giving further detail.

10.8.3 How Deep?

The depth of a simulation is the number of moves that you look ahead along each variation. The goal is to allow tactical, short-term factors to come to light.

While it is possible for factors to take several moves to work out, it usually requires just two plies in Scrabble. Scrabble racks turn over often. Average moves turn over 4.5 tiles, so after a couple of moves there is usually nothing left from the original rack. Hence, two plies is generally ideal.

Still, sometimes 4 plies of lookahead show a different picture than two. Some players routinely simulate to four plies for that reason.

One must be wary about using deeper simulations! They take more iterations to reach statistical accuracy and they are slower as well. For example, a four-ply lookahead has double the variance of a two-ply simulation, and each iteration takes twice as long. Therefore, the CPU cost is tripled when using a four-ply lookahead. We maintain that if it requires an extreme situation to see the value of four-ply simulations then they are not worth doing.

We have seen that even in the extreme case of a move like the opening rack BRONZER (8H) that leaves the largest conceivable opening on the board, a two-ply simulation is still able to determine the point differential to within a point. If so, then there is no value in using deeper simulations. Other analysts disagree (particularly Joel Sherman)[70], but that is my opinion and I am sticking to it.

One key exception occurs when the game ends along one variation. If it is possible for the game to end within the normal search horizon then every variation must be searched to the end of the game. The reason for this rule is that one of the foundational assumptions of simulation is violated: that future turns (which are not simulated) are equal for all positions, at least on average.

The point is that the Basic model does not include any bonus for the right to move, because the same player (the opponent) is always moving next. There are small positional bonuses, but these are present only to account for the *differences* between the right to move in some positions rather than others. The *absolute* value of the right to move is not accounted for anywhere.

The absolute value of the right to move is large, because the side to move first has about a 50-50 chance of moving last, which would give him an extra turn for the game as a whole. The typical value may be in the range of 15 to 20 points, depending on the position.

Let us return to considering what happens when a simulation variation includes the end of the game. The opponent, who would normally be on move in that situation, will have a future value of zero points, because the game is over. In contrast, along other variations the opponent actually has the right to move when the variation ends, and therefore stands 15 to 20 points better off than he does when the game ends. Since this difference is not accounted for, a significant amount of inaccuracy is involved.

You can try to repair this defect by including a right-to-move bonus, and that would significantly improve the accuracy. However, since this situation only occurs towards the end of the game, why not spend a little extra CPU time and benefit from obtaining a winning percentage estimate? It costs relatively little and adds a lot to the program's skill.

10.8.4 Time Control in Simulation

There are about 12 moves per side per game on average, but you need to consider that even the longest game should be complete within tournament time control. The longest games are about 16 moves per side.[45]

Tournament time controls are 25 minutes, but for a computer program you need to factor in time for the operator to handle the physical transfer of tiles from bag to rack, and from rack to board, plus typing the opponent's moves and filling in the official scoresheet. These functions take about 5 minutes per game. Therefore, you have only 20 minutes for thinking.

Then there is the issue of leaving time for unexpected events. For example, what if the operator messes up entering a move? What if a power spike crashes the computer? What if the program holds the blank for 8 turns and needs extra time to think? Out of respect for these possibilities, a practical application of simulation to Scrabble requires that moves average about 1 minute of thinking time.

With each iteration looking ahead 2 plies after 10 candidate moves, and 1000 iterations per turn, we would have 20,000 move generations. When the author began work on the 1998 match against Logan, MAVEN could play 20,000 moves in 250 seconds. This is over 4 minutes, so we needed to cut back somewhere.

Reducing the number of moves or iterations is possible, but these changes increase the error rate. Of course, if the goal is simply to improve upon the play of the Basic evaluator then we do not need much; for that lowly goal, we need only simulate *two* hmoves. However, if the goal is lofty (i.e., almost perfect play), then we should not reduce quality.

Fortunately, we have a resource that allows the move list to be pruned with limited risk. We can prune any move that falls N standard deviations below the best play. That rapidly cuts the number of trials. For example, taking a look at the distribution of differences between first and second moves, we see that almost 40% of turns have a best play that exceed the second best by 4 points or more. With a rejection level of 2 standard deviations, such simulations would quickly terminate the search with a unique best play after only 500 iterations. Back-of-the-envelope calculations showed that by using this

[45] Human games can be longer, but I have no recorded MAVEN games longer than 16 turns.

optimization we would be able to pick up almost the entire factor of 4 needed to play in real time using simulations. The remainder we made up by running on a faster computer during the competition.

10.9 Practical Implementation

This section describes the actual implementation used in the 1998 match. The implementation was not ideal, but was is effective and illustrates important principles that apply to stochastic search controllers.

10.9.1 Rack Sampling in General

It is important to reduce the variance of comparisons between moves by using parallel racks. That is, each iteration uses identical racks across all trials in that iteration. Since the racks are the primary determinant of the score of a trial, you greatly reduce variance with this technique. The only exception is if there is an exchange of tiles in the move sequence, but exchanges are rare. In practice, the tiles will come out in the same order on every trial.

Another key technique is to aggressively enforce the desired tile distribution. For instance, in the 1998 match MAVEN always used a uniform tile distribution. MAVEN ensured that the distribution was as close as possible to uniform by seeding each iteration with the one tile that had been most underrepresented in the racks thus far.[46] This slight pressure to move towards uniform is all that is needed to ensure that tiles are distributed fairly. This trick ensures that the moves are not biased by an unusual frequency of draws from the bag.

It has been misreported that MAVEN attempts to give opponents a well-balanced rack [71]. This is probably a misinterpretation of what it means for a rack to be sampled from a uniform probability distribution. The tiles occur randomly with equal probability, and MAVEN takes pains to ensure that they actually do so in the samples it simulates. However, the term "well-balanced" means something different to a Scrabble player, who would normally take it to mean "having an equal number of vowels and consonants." Scrabble players are often surprised to find out that simulations use uniform racks rather than balanced racks, because the racks that occur in games are decidedly not uniform, and tend to be balanced.

10.9.2 Enumerated Endgames

When the number of tiles in the bag is small then MAVEN exhaustively enumerates all racks. The standard is that if the number of distinct racks is less than the maximum number of iterations and all moves empty the bag then exhaustively enumerate. If there are 12 distinct tiles in the bag then there are 792 distinct racks, which is well within the limit of 1000 iterations. Of course, the number of distinct racks may be lower because of duplicate tiles, and some rack may occur with higher frequency. The simulator must account for these effects.

[46] There is a simpler algorithm for ensuring a uniform distribution. The algorithm is simply to draw tiles from the bag *without replacement* until the bag is depleted. This ensures that the tiles occur with uniform frequency. I decided on the trick of seeding the most underrepresented tile because it generalizes to non-uniform distributions. In the year 2002, that 13-year old design decision is finally paying off, as I begin experiments using inferences.

As a special case, MAVEN used the endgame search engine to analyze the full endgame if the position was a PEG-1 and the move emptied the bag. This endgame analyzer was modified to stop searching once it proved that a variation was either definitely winning or definitely losing. The endgame engine employed a time limit to prevent unlimited thought. Since the evaluation function of the endgame search engine is an interval, and simulation requires point estimates, MAVEN needed a conversion rule to adapt the result of the endgame search to the need of the simulator. The algorithm was to return a bound that proved the win or loss, if such a proof was achieved during the search, or to return the average of optimistic and pessimistic bounds otherwise. When MAVEN returned the midpoint estimate, it estimated winning percentage by interpolating the point differential required to win within the endgame engine's evaluation interval.

10.9.3 Search Horizon

We have indicated the answer to the question of search depth: the normal search depth is two plies. This means that we place the move and evaluate by using a sample of variations of length 2. The exception is in the pre-endgame, when we search to the end of the game. We have described how the possibility of a game ending during a variation should be avoided.

10.9.4 Number of Moves

In midgame positions, MAVEN searched 10 moves. This policy includes the best play between 90% and 95% of the time.

Pre-endgames are more complicated, and MAVEN searches more moves. MAVEN used four move generators (Basic, pre-endgame, ultra-paranoid, and score maximizing). It is possible for the engines to generate as many as 40 moves, but MAVEN seldom sees more than 20 because there is great overlap between the lists of moves.

One of the great things about simulation is that it is able to compare moves that were evaluated by different evaluation functions. For example, suppose that in a pre-endgame the Basic evaluator considers move A to be best, but the pre-endgame evaluator prefers move B. Which move is better? The answer is probably B, but it may be A. Simulation provides an answer. MAVEN can simulate games after moves A and B, and then compare the moves based on point differential or winning percentage.

10.9.5 Pruning Rules

The simulator always stopped after 1000 iterations. In a pre-endgame where MAVEN enumerated all of the racks and all of the moves empty the bag then MAVEN stopped after going through all racks. Note that if any move does not empty the bag then you must continue sampling because the side-to-move could draw different tiles.

Simulation stopped if only one move remained.

MAVEN cut any move that differed from the best by at least two standard deviations, but MAVEN never pruned a move before the 17th iteration, because we wanted the data that measures the standard deviation to develop before risking any cuts.

Once a move was pruned, it was never searched again, even if the score of the best move changed. MAVEN's pruning rules were sufficiently reliable that there was no benefit to reintroducing moves. For instance, the 14 games of the 1998 match had only one instance in which the best move was pruned, costing MAVEN only a couple of points.

It often happens that two plays use exactly the same tiles, play in exactly the same spot, and score the same. In such situations, there is usually not an iota of difference between the plays, so a simulation would go to 1000 iterations only to determine that there was no reason to prefer either play. Therefore, to avoid this inefficiency MAVEN arbitrarily eliminated the lower rated alternative at iteration 31.

It often happens that there are multiple legal bingos. Two bingos always have the same rack leave, and they often have the same score. It is usually unimportant which bingo is selected, so MAVEN arbitrarily eliminated the lower rated play at iteration 100.

Once play reached the pre-endgame, MAVEN measured two dimensions: point differential and winning percentage. The idea was to choose moves that had the highest winning percentage, but you cannot ignore point differential. There are several reasons why not.

1) Winning percentage is often equal, especially when one side is overwhelmingly ahead. Then it is proper to maximize point differential.

2) Winning percentage is often approximately equal, even when point differentials are significantly different. Then the move with the higher point differential has a significant practical advantage: the opponent is likely to make an error that boosts the winning chance of the higher-scoring move by a relatively larger amount.

3) Ties between tournament participants are broken by point differential, so racking up points is beneficial even if a small amount of winning percentage is lost. In fact, most players are tied with someone at the end of the tournament, so winning on point differential is as valuable as tying an extra game.

MAVEN's strategy was to prune moves using winning percentage and point differential on equal terms, according to the following rules:

1) If point differential was 2 standard deviations worse and winning percentage was not better then prune the move.

2) If winning percentage was 2 standard deviations worse and point differential was not better then prune the move.

Thus, at the end of the simulation all of the remaining moves had no significant disadvantage with respect to point differential or winning percentage, or they are worse in one but better in the other. MAVEN chose the move that maximized a linear combination of winning percentage and point differential. The ratio between winning percentage and point differential was set so that 100 points of differential equaled ½ game. The author computed that tradeoff by looking at the standings in the 1990 North American Championship.

10.10 Limitations of the 1998 Implementation of Simulation

The simulator implementation used in the 1998 match had shortcomings. Some are simple bugs. Three specific bugs are

1) MAVEN never generated any exchanging moves. Oops!

2) When MAVEN actually got into an endgame position, the endgame player stopped searching once it found a proven win, just as if it were searching the endgame variations of a PEG-1. Oops!

3) A two-ply simulation can see through to the end of the game when there are 21 unseen tiles. Simulation to the end of the game should have commenced at 21 tiles unseen, but actually commenced at 16 tiles unseen. Oops!

Such bugs were easily fixed, but other shortcomings were fundamental design issues.

The biggest failing is that the implementation is not scalable. It would be nice if the simulation controller took advantage of faster computers in multiple ways. For example, on a faster computer the program should be able to search more moves for more iterations and make its decisions in less elapsed time. In the 1998 implementation, the behavior of the controller was fixed by limits that did not depend on the speed of the host computer. Therefore, the only benefit gained from a faster computer was lower elapsed time.

To be clear: MAVEN *would* benefit from faster play. Humans are accustomed to playing at a measured pace. Computer programs should disrupt that steady rhythm by slapping down moves as fast as possible. Ideally, MAVEN's move would be on the board before the opponent has finished drawing his tiles. That is possible when MAVEN is not using simulation, and it would be good to get that capability back.

The other side of scalability is to benefit from simulation even if running on a slower computer. MAVEN's 1998 controller would not run satisfactorily on less than a 300 MHz Pentium II. The author believes it is possible to perform simulations with *some* benefit even on a 486/66. The controller might only be able to look at two moves, and then only for a few hundred iterations, but it would find a play or two per game.

Many people regard MAVEN's selection of random racks to be a weakness. For example, consider Graham Toal's proposal to select high-probability racks instead [71]. There is real risk in trying to choose a set of racks. The goal is for the racks to accurately represent the distribution of the opponent's possible holding, and it is not clear how to do this with a deterministic selection of racks. There are two objections to MAVEN's procedure, one on the grounds of efficiency and the other on the grounds of accuracy.

Ron Tiekert believed that simulation could be much faster if a scientific selection of, say, 50 racks could be drawn that essentially represent the full conceptual space of Scrabble [61]. For example, the sample would have a number of bingoish racks, a proper number of racks with only consonants, the right number of Q's, and so on. The concept is like trying to find an "orthonormal basis" of order 50 for the space of all Scrabble racks.

Other critics suggest that drawing random racks is clearly wrong because racks in real games are better than random. Should MAVEN use a distribution that actually applies in real games? The author is not sure this matters, because of cancellation of errors; the same distribution applies to all moves being simulated, so MAVEN will probably choose the best move anyway.

Rather than pursue what may be a mirage, MAVEN draws racks randomly. In the author's opinion, improving on this policy is hard, because it is difficult to make a robust inference engine. Chapter 11 covers the subject in detail.

Chapter 11 – Potential Improvements in Simulation

This chapter discusses opportunities in the area of simulation. Simulation has been successful in MAVEN, but there remain technical and modeling deficiencies. In addition, there simulation creates new opportunities, as well.

The first section describes how to infer the contents of the opponent's rack by carefully considering the opponent's recent moves. Simulation can then exploit those inferences by biasing the distribution of racks held by the opponent.

The second section concerns how we can exploit the weaknesses of the opponent by creating a model of his play. Given such a model we can bias simulation to select moves that specifically defeat the opponent.

The third section describes how to use simulation to select moves that have the highest winning percentage. It seems obvious, but there are complexities.

The fourth section concerns the speed of simulation, which is vitally important if you want to put time pressure on the opponent.

Finally, simulation gives us a means of evaluating phonies. The last section concerns how to generate moves that include phony words, and how to use simulations to evaluate whether the gain is worth the risk.

11.1 Inferences

In the current model, the distribution of the opponent's unseen tiles is assumed to be uniform (i.e., every tile equally likely). However, we can determine from the opponent's previous moves that certain tiles are more likely than others are. Does knowing this type of information change the play?

Let us take an extreme example. Your opponent plays JANTIES. You think to yourself, "Humph. My opponent has been studying SOWPODS too much, which has obviously addled his brain." You challenge. Then JANTIES (a word in the OSW but not the OSPD) comes off the board. In this situation, you know exactly what the opponent held, and this has the potential to radically affect your play. For example, you would not want to hang a U in open space, since it would set up the OSPD-legal JAUNTIES for the opponent.

To continue the example, suppose that after your play the opponent plays TAJ. While you do not know what the opponent drew, you do know that he held TAJ-EINS before his move, so he must have EINS still on his rack. More precisely, his rack consists of EINS plus 3 tiles drawn from a uniform distribution. It is hard to say how this would affect your choice of move, but opening bingo lines is probably somewhat worse than normal.

To continue the example in a different way, suppose that after you challenge JANTIES, the opponent's next turn is to exchange one tile. All you know for sure is that the opponent retained six of the JANTIES tiles. Nevertheless, you can realistically eliminate all but exchanging the J. The tiles AEINST make a bingo with 90 tiles from a full bag, as only J, Q, and Y are "nongos." No one with any sense would keep the J in that situation.

This requires a more complicated inference about the opponent's rack, but you can till make a strong statement about the true distribution of the opponent's tiles.

Inferences have a role in some of the most amusing war stories. In human-human play, savvy players often draw conclusions based upon the opponent's demeanor. One story involved Charlie Southwell noticing that the tiles on the opponent's rack were split into a group of 4 tiles and a group of 3 tiles. Whereupon Southwell found a spot where an 8-letter bingo fit with four tiles before contact with the board and 3 tiles after. Southwell blocked the spot, and it turned out that the opponent actually held the necessary tiles. However, the author is not about to start work on a machine vision system.

More intriguing was Joe Edley's observation that you can learn things from the opponent's sequence of scores. This is especially true of players in the 1800 to 1900 range, since they have the ability to find high-frequency bingos, but generally will miss low-frequency bingos. Accordingly, such players tend to make a series of moves that dump awkward tiles and retain ADEILNORST to build a bingo. Edley maintains (and I believe, though it is hard to disprove) that he can *feel* when his opponent is about to bingo. MAVEN's moves do not seem to have such "tells" in its play, and true masters (NSA rating >= 1950) do not telegraph their racks either.

Of course, there are other signs. For instance, most humans will not leave a hook in the triple-word column unless they hold the hook tile. Accordingly, humans have a strong tendency to block such hooks created by their opponents, on the inference that if the opponent made the spot then he must hold the tile. However, MAVEN makes such spots with abandon and on balance is not hurt by this habit. Humans may eventually "unlearn" this particular inference.

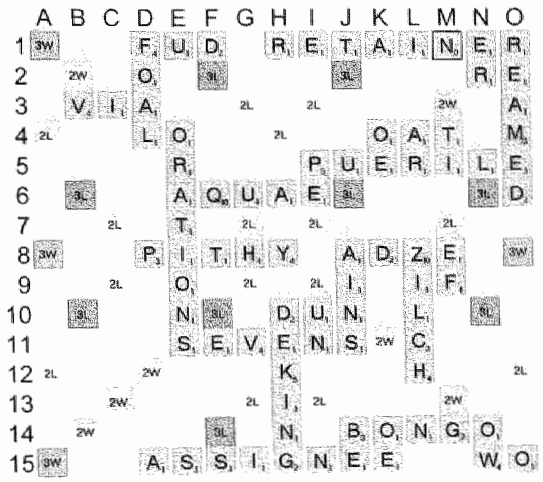
A more solid conclusion is that if the opponent played a U then he did not keep a Q. This is almost ironclad. However, it is hard to see how such an inference helps you to make better plays.

Another example came from the author's first tournament. The author is a good player (NSA rating 1840) but had never played against humans before; only MAVEN. One friendly player sat me down with the idea of showing me the lay of the land. We played a game, and at one point, she made a strange play. It scored only 7 points, and I was sure that it was a blunder. The opponent's next move was to play a 50-point X play overlapping that low-scoring move. My opponent explained that human often create such setups, and since I had only played against the computer I would be unaware of such tactics. Her point was that since I had been alert to the strangeness of the move, I should have looked one step beyond to find out why she played it. There had to be a reason! Indeed, had I thought about it a second then I would have seen the potential X play, since I could not miss such a thing if I think about it. She was correct; I probably left some equity on the table. But clearly she hurt herself with the two-move setup. She scored 57 points over two moves, which works out to just 28.5 per turn. When you consider that she burned an X to accomplish this, you have to figure that she hurt herself on balance. How logical is it to infer that she set out to hurt herself? If one concludes that the opponent is more likely to have an X, how many points can one sacrifice to block the spot? Her average move using an X is probably 38 points, so if I were certain that she held an X then I could justify a sacrifice of up to 12 points.

So the author concluded, in agreement with Nick Ballard [62], that humans overdo such considerations, particularly at the intermediate level of skill where a little knowledge is a dangerous thing. Players at that stage hurt themselves by playing for such setups. Then their opponents justify the action by imagining threats and playing to block them, often sacrificing many more points than the opponent would have gained had the threat been carried out.

The scientific study of inferences began when Nick Ballard published an extraordinary monograph entitled *Anatomy of an Endgame* [63] in which he brilliantly analyzed the best play possibilities after an opponent's fish. The subject of the monograph is Position 11-1. Though there are 92 distinct racks using the unseen tiles, inferences from the opponent's last play enabled Ballard to conclude that the opponent's last rack must have been ?AEILRR, ?AELRRT, or ?AERRTT. The logic was as follows:

1) The opponent could not have kept J or W and still have a proper fish.
2) The fishing the R is normally poor unless the opponent held RR.
3) Opponent would have played a bingo if he had one.



Moving:	B, C, G, M, O, X, Y,	340
Opponent's last:	RE (2N,8)	322
10 Unseen tiles:	?AEIJLRTTW	

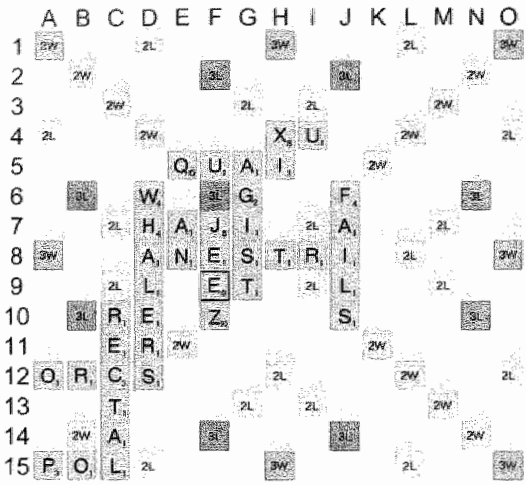
Position 11-1 Ballard's Brilliant *Anatomy of an Endgame*

Ballard could then break the analysis into cases according to the opponent's draw from the bag. There were nine cases. In each case, Ballard considered each of the four candidate plays, and then fully solved the resulting endgames. Actually, the best move (MIX (13G, 23)) does not empty the bag, which means that the analysis required an additional ply of search in that variation.

Ballard carried out this analysis, which ran to 26 pages, without the benefit of a computer. When MAVEN checked Ballard's analysis, the author was astounded by the degree of accuracy of the calculation. This *tour de force* established that inferences were a potentially huge benefit.

With the aid of MAVEN, the author was able to perform an inferential analysis of a complicated position, shown in Position 11-2 The reader might recall seeing it as an example of the power of simulation, which discovers the well-hidden strengths of the move AY (D14, 24, CIOPT). Now the question is whether we can draw any inferences from the opponent's last play, POL (15A, 23), which played the tiles P, O, and L.

The opponent's rack contained the tiles POL plus 4 of the unseen tiles. 6900 racks contained the letters POL, but the author was able to prove that the move POL as played was reasonable for only 50 of those possibilities. (The fact that the opponent's actual rack leave was among the 50 did wonders for the credibility of this assertion.)



Position 11-2 What Can We Infer from POL?

The standard used in determining the possible racks was that MAVEN had to consider POL to be one of the three best plays in the position, and consider POL to be within 5 points of the best play. We also eliminated any rack in which POL was manifestly inferior to another move. Had we only allowed those racks for which POL was the top play (i.e., as if the opponent were MAVEN itself) then the inference would have been further restricted.

I manually generated possible racks, then used MAVEN to generate moves, and then manually tracked the results. My first conclusion was that the opponent did not hold any of EHLOUWY, because of the plays POLE, HOLP, POLL, POLO, PUL, OWL and POLY, all played at 15A and all yielding an indisputably superior combination of points and rack leave. In addition, the opponent could not have retained G, because of GALOP (14B, 28).

Next, I eliminated many pairs of tiles. For example, the words from Table 11-1, play at 15A, and each eliminates a pair of tiles that the opponent could have kept.

There are other ways to eliminate possibilities. For example, from B11 we find FRAP for 34, so AF is impossible. On the A-column, we have a few plays that help: BLOOPS and POOFS eliminate BS and FS, PROMO (A8, 27) eliminates MR. VAPOR (14B, 24) eliminates RV, and VAV (14B,18) eliminates VV.

Word	Tiles
BOLA	AB
BOLT	BT
DOL	DD
FILO	FI
FOLK	FK
PILOT	IT
KILO	IK
MOLA	AM
MOLT	MT
TOLD	DT
SALP	AS
POLAR	AR
BOLD	BD
MOL	MM
PAL	AA
FOLD	DF
PILI	II
POLKA	AK
MILO	IM
MOLD	DM
SOLD	DS
VOLT	TV
POLIS	IS

Table 11-1 Moves at 15A that Eliminate Pairs

While there are exactly 6900 racks the opponent could theoretically have kept, only a few hundred match our inferences. Simply removing EGHLOUWY cut the number of possible racks down to 948. When the pairs of tiles were removed, the remainder could be completely enumerated by a human. In doing this work, it is helpful to note words that help eliminate further possibilities. For example, VANPOOL (O7, 36) eliminates all possibilities containing ANV.

Little was left. I fed each remaining combination into MAVEN to get an opinion on the move. Though there were a few hundred combinations, POL was a reasonable play in just 42. The 42 combinations are in Table 11-2. In making this list I had to decide what to do with the ?K combination. POLKA (15A) scores 53, where POL scores 23. POLKA is usually best, so I am not listing the combinations in which K and blank occurred together.

?BBF	?BFV	?MNV	BFMV	BKNV	FMNV
?BBM	?BIV	?MSV	BFNR	BMNN	KMNV
?BBR	?BMN	BBFM	BFNV	BMNV	KMSV
?BBV	?DNN	BBFR	BINN	DINR	MNNS
?BFM	?DNV	BBFV	BKMN	DKNR	MNNV
?BFN	?FNV	BBMV	BKMV	DKNV	MNSV
?BFR	?MNN	BFMN	BKNR	DNNV	NNSV

Table 11-2 All Normal Racks Opponent Could Hold

Let us take another tack, just to make sure we have covered all possible racks. Could the opponent have had a bingoish rack, and dumped POL to keep great tiles? For which racks was that a reasonable strategy?

Clearly, the opponent must have held good tiles: ADEINRST, with no duplicates. Also, the opponent would only fish if he got excellent bingo equity in return for his risk, because he is ahead and he could score 6 to 9 extra points at 15A. One deduction is that the opponent could not hold 36 combinations, because they lead to bingos when combined with POL. I then eliminated combinations having the DE pair, because POLED adds 12 points, which seems like too many point to justify a fish. There remained 29 possibilities. MAVEN generated each to determine if a reasonable case could be made for a fish. Only ADRS, AIRS, DINS, DIRS, DNRS, EIRS, ERST, and INRS offered reasonable fishing prospects:

The opponent held, if not exactly one of the given racks then at least one that is similar. (Or the opponent made a big mistake!) It is reasonable to calculate our replies assuming the opponent's rack was one of these.

There are 42 normal racks plus 8 fishing racks for 50 cases, but the cases are not equally likely. The distribution of tiles favors holdings like DINR, because there are 4 Ds, 5 Is, 5 Ns, and 2 Rs, and makes holdings like BBFV unlikely. To account for the distribution of tiles, we must weight each case by multiplying together the number of tiles in the bag of each type used in the leave. For example, DINR has a weight of 4 * 5 * 5 * 2 = 200, which is the largest weight. The second largest weight is EIRS with weight 100. BINN (the opponent's actual holding) has weight 50. When each case is weighted according to its likelihood, there are 1313 cases.

The chance the opponent holds the blank might surprise you. Blanks occur in 188 weighted cases, for about a 14% chance. Additionally, the opponent draws the blank with probability 3/49 when he does not already hold it, so the total chance that he holds the blank is about 20%. This is higher than we would expect based upon chance; normally we would expect him to hold a blank only 7/53 = 13.2% of the time.

Curious: when the opponent retained a blank from last turn he has virtually no chance of playing a bingo next turn. Thus, there is temporary tactical advantage to keeping an open board. However, most of our own rack leaves are not bingoish, which diminishes this advantage.

This result was eye opening. These calculations demonstrated a reasonable procedure for inferring the distribution of unseen tiles, at least on the assumption that the opponent was master-class. Moreover, the reduction in the size of the search space was enormous; from 9600 down to 50 makes concrete calculations realistic. What's more, the racks could be further grouped on an abstract level into "consonant-heavy" racks, "blank plus drek" racks, and fishing racks. In the mind of a human master, the branching factor was just three!

In addition, for the first time it was shown how to use inferences in simulations. The author used a spreadsheet to re-weigh MAVEN simulation data according to the frequency in the inferred distribution. Though the calculation was manual, the path to computerization was clear. Some MAVEN users have developed postprocessors that munge MAVEN's simulation log files to compute evaluations under a non-uniform scenario. John Chew's POSLFIT has some ability to do this [64].

While manual calculations and postprocessors suggest that full automation of inferences is close at hand, full automation will require solving some difficult problems.

The first question is whether the model given above is the best choice. There are alternatives. For example, Alan Frank followed a suggestion of John Babina, and published a model in which the score and turnover of the opponent's last play indexes a table of information about the opponent's likely holding [65]. This approach does not have the "X-ray vision" potential of the algorithm sketched above, but it is less likely to be fooled badly. Frank's suggestion has convinced me of the value of a decision tree that branches based on easily discernable aspects of the situation before engaging in detailed analysis. For example, if the opponent plays one tile it could mean he is trying to fish a bingo, or it could mean that he is trying to avoid drawing the Q. Simply knowing whether the Q is in the bag could help to distinguish these cases.

Another open question is how to represent inferences. Very naive representations are probably excluded for performance reasons, since there are over 3 million possible seven tile racks. However, representations that are not capable of representing any possible combination of tiles may lose precision. What level of detail is required?

There is an additional desirable feature. In the pre-endgame there are few distinct racks, so it is sometimes possible to enumerate all of them. It would be nice if the solution could supply this service.

Another desirable goal is to be able to subsume the uniform distribution without a special case, preferably without an infinite number of buckets.

Given a probability distribution, another open problem is to generate racks for simulation purposes. There are two extreme solutions. One extreme is to generate racks without duplication, and then reweigh the results according to the distribution. However, the convergence properties of this approach are poor, because the sample has relatively few instances from the highest-frequency racks. The other extreme is to generate samples in proportion to their likelihood, but then it is hard to match the expected frequencies. This is a complex striated sampling problem.

A representation scheme will need a generalization of MAVEN's trick that enforces a distribution. In this sampling problem, the buckets are the possible racks that the opponent could have held. For example, buckets ABC and ADE might be possible, with relative weights 9 * 2 * 2 and 9 * 4 * 12, respectively. A sample is drawn from a bucket by filling in the known letters (e.g., ABC) and then drawing uniformly from the rest of the bag. Suppose that the draw is DEFG, making a trial of ABCDEFG. The trial also contains the tiles ADE, and so it is a trial for both bucket ABC and for ADE. The weight of the rack ABCDEFG within the bucket ABC equals the chance of drawing DEFG from the bag, which differs from the weight of ABCDEFG within the bucket ADE (i.e., the chance of drawing BCFG from the bag). Does this matter?

If the foregoing machinery all works, then you have the ability to exploit inferences. Now, how do you generate inferences?

The manual calculations described earlier suggest that a direct approach will work. However, the scalability of the technique is a concern. For example, the case of POL,

above, featured only four tiles left on the rack. Would the calculation be feasible with more tiles left? Would even a four-tile leave be feasible if we were unable to eliminate many tiles in advance? How many move generations are needed before the computation is complete?

How will you handle uncertainty in the calculations? Should you assume that a human would have found an 11-letter bingo had one been available?

How would you handle contradictions? For example, suppose that the opponent's tiles obtain a higher score elsewhere. Would your engine reject all racks, leaving us with no buckets, or with all buckets of weight zero?

How would you handle generic inferences? For example, MAVEN's games show that if a blank is unseen in a PEG-1 and the opponent played two tiles last turn, then the blank is in the bag about 6%, rather than the 12.5% you would expect from the uniform distribution. Note that there is no specific move that justifies this conclusion; it is just a statistical result.

How would you validate the system? Can you check your inferences against Internet games? Can you check your inferences against the play of your own engine? How accurate are your conclusions when drawn against your own simulator? That question is interesting because the simulator would play more strongly than the inference engine would.

One open question about inferences is how to draw inferences when playing weak opponents. You should not assume that weak opponents have excellent move generation skills. Is it still valid to fall back on certain types of inferences? Is it valid simply to make inferences as if the opponent were strong, because if he plays weakly then we will win anyway? Should we "mix" the inferred distribution with a uniform distribution at a rate that reflects the opponent's skill level? Should we just disable the whole thing?

Here is a story from Robert Felt. Felt was playing a weak player, and having a tough time. Every master experiences games like this, where every draw from the bag is bad and the game reaches the end with a weak opponent poised for an upset. This situation was particularly complicated by the fact that the Q was unseen, and the opponent's last turn was to exchange six tiles. Obviously, after the opponent's exchange the Q must be in the bag. Therefore, Felt went into the standard technical approach to this situation, which is to play one tile at a time in the hope that the opponent draws the Q. The opponent, however, seemed oblivious to this tactic, and played several tiles per turn. Felt's happiness was short-lived, however, when he discovered that the opponent had *kept* the Q when she exchanged. Her "idea" was that the Q was worth 10 points, so she considered Q to be a good tile and wanted to draw the last U to go along with it. In this scenario, Felt's expert tactics were inappropriate, and backfired predictably. Ouch!

By far the biggest open question is whether inferences are worth doing at all. The author has calculated a few of these situations by hand, and the move decision rarely changes. Actually, the trend has been for the best move under the uniform distribution to have an even bigger relative advantage under the non-uniform distribution. A case in point is our example, where the move AY has an even larger advantage when inferences are considered. Recent large-scale computer investigations suggest that inferences improve

results by at most a couple of points per game, but these results are preliminary and I will refrain from giving further details.

11.2 Opponent Model

Simulation is one of the best ways of including an opponent model into a program. If you can create a program that has the same responses as an actual opponent, then you can use simulation to devise tactics that specifically defeat him.

For example, Charlie Carroll likes to play words that take front hooks [73]. His experience is that non-masters tend to miss the opportunity to use such hooks. Even masters miss them! They are also likely to miss back hooks for non-S tiles. What is the benefit from creating such hotspots? Simulations would tell us, to the extent that an opponent model captures the weakness.

Another example is to exploit MAVEN's advantage in finding bingos. We can measure this difference as a function of the bingo-openness measure defined in Chapter 6. The difference is perhaps greatest on eight-letter bingos. For instance, the author (NSA 1840 rating) plays 8-letter bingos and 7-letter bingos in 1 to 1 ratio. A sample of master games shows that strong masters play 1.7 times as many 8-letter bingos as 7-letter bingos. MAVEN plays 2 times as many 8-letter bingos as 7-letter bingos. It follows that controlling the opportunities for 7-letter bingos has a greater impact on anti-human tactics. Again, an opponent model that missed more eight-letter bingos would allow simulation to discover this behavior.

Can we build a realistic model of weaker opponents? We probably can. MAVEN has taken some steps along this path, and a great deal more can be done.

MAVEN has a system that scales play down to lower levels. MAVEN's first effort simply involved throwing out moves randomly. This technique provided good scaling down to about the 1000 rating level, but suffered from a number of defects. First, the 1000 rating level turns out to be excessively strong for casual human opponents, who are more like 700 rating. Second, the scheme had unfortunate side effects. Three specifically stand out. First, when MAVEN's tiles were bad it could randomly skip all of the moves, which lead to the program exchanging too often. Second, after the program exchanged it usually got a good rack, and then landed a big score. Thus, the pattern was weak overall, but consisted of a mixture of zeroes and big scores. Another defect was that the program's vocabulary at the weakest levels was the same as the highest levels, which is most unnatural.

MAVEN's second effort involved creating a vocabulary of common words specifically for the lower levels of skill. This task required input from many people, since the author's judgment of whether a word is common is thoroughly corrupted by long association with MAVEN. Over time, this eliminated the vocabulary mismatch issue. I also discovered that MAVEN could play at the 1800 rating level without ever using an uncommon word, so I raised the level at which MAVEN employs the tournament vocabulary.

MAVEN's third effort involved selecting words according to their percentile rank, which was more stable. With this system, MAVEN could play all the way down to 700 rating without discontinuous behaviors.

Nevertheless, there remain differences between how MAVEN behaves and how weak humans behave. At its lower levels, MAVEN is simply matching the scoring characteristics of human players, but not other aspects. If you want a simulation to exploit human weakness then you probably need more than this.

We can formalize the model as follows: given a board and the best move on that board, return the probability that a human of a given level of skill will select that move. If you have such a function then you can emulate a human as follows:

1) Generate the list of moves in best-first order.
2) Loop down the list computing the probability of selection, and
3) Select a move in proportion to its probability, choosing the highest-ranked move that is selected.

A neural network would provide a good model, if it had inputs that reflected the difficulty of selecting the move. Consider the following inputs:

1) Common versus uncommon word. (Vocabulary.)
2) Length of word. (Anagramming.)
3) Uses JQXZ? (Anagramming.)
4) Is a hook? (Anagramming. Obviousness.)
5) How many words formed by the play? (Move complexity.)
6) How many legal moves in total? (Board complexity.)
7) Is the player leading at the time? (Attentiveness measure.)
8) Frequency of word in MAVEN versus MAVEN games. (Vocabulary measure.)
9) Number of letters used from the board. (Move complexity.)
10) Premium squares covered. (Obviousness.)
11) How many moves have been played? (To get at tiredness, board complexity.)
12) Prefix + Suffix length (Anagramming.)
13) Uses blank (Anagramming.)

The training data for the model would come from Internet games. There are several Internet Scrabble game rooms, where you can spy on games to collect data. Over the course of a year, you would collect tons of information about how intermediate-class human experts play the game.

Validating an opponent's model is tricky. It is a sort of Turing test—can you distinguish a human of a specific rating from the computer simulation? Well, unless you have programmed the simulation to play phony words on occasion you will be able to distinguish human from computer. However, that exception is but a small obstacle.

There are many tests of validity. Simplest is to determine whether the model plays with the proper level of skill, which is a simple matter of setting up a tournament between the levels and rating the outcome.

If the players perform at an appropriate level, then you can test whether they are able to mimic the playing characteristics of humans in online games. Simplest is to compute the model's probability of selecting the moves made by humans of an appropriate rating. The

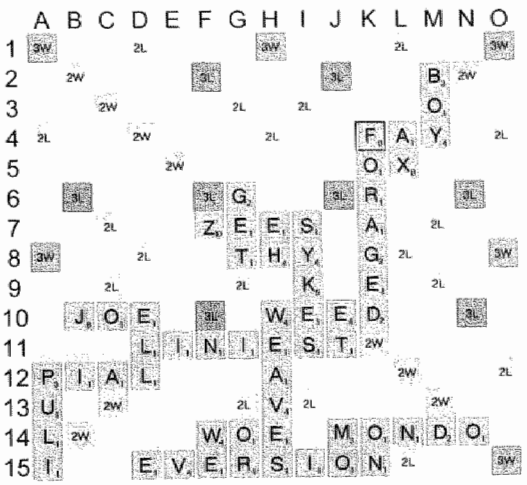
model that most closely matches the actual moves should be the model that was calibrated from games of players of that rating.

It is harder to determine whether an opponent model helps MAVEN to win more often. You can test MAVEN in online games by employing an opponent model in half of the games. It might take many games to see a difference. You can test MAVEN versus the model directly, but that might be a self-fulfilling prophecy.

11.3 Winning percentage

MAVEN's 1998 implementation collected winning percentage data near the end of the game, but it is potentially valuable to gather such information throughout a game. Look at Position 11-3 from MAVEN's match against Adam Logan. In this position, MAVEN played FOP (N1, 28, AEGT), leaving an overlap spot for a bingo on the O-column. After the game, Adam commented that FOP was a dangerous play. He recommended OFT (15M, 26, AEGP), which blocks bingos to MONDO.

MAVEN was attracted to FOP because FOP sets up the AGE that MAVEN keeps on its rack. AGE (O1) scores 39, and there are many draws from the bag that can improve it. It is not a one-way setup by any means, since if Logan held an A he would usually be able to exploit the spot, but Logan is less than 50% to hold an A, and even if he does he may not be able to use the spot.



MAVEN:	A, E, F, G, O, P, T,	287
Logan's last:	PULI (A12, 18)	244

Position 11-3 Winning Percentage versus Point Differential

Logan's view of the situation differed. Logan considers that MAVEN is 43 points up before the move, and will take a 70-point lead. If Logan landed a normal bingo (e.g., 75 points) then MAVEN would still be a favorite. For example, if Logan played across the second row through BOY, with MISCIBLE (8H) for instance. MAVEN starts out leading by 43, and OFT puts MAVEN up by 69. MISCIBLE scores 82 points, so MAVEN regains the move down by only 13. Since 13 points is less than half a turn, MAVEN would still be a favorite. That is particularly so because MISCIBLE would open up two triple-word squares at O1 and H1.

What actually happened in the game is that THENARS (O3, 97) followed FOP. When compared with the example of MISCIBLE, THENARS scored 15 extra points, and does not yield openings that MAVEN could exploit. In fact, the board was devoid of spots for large comebacks, and MAVEN was unable to close the gap.

MAVEN has never done a winning percentage evaluation of this position, so we can neither confirm nor deny Logan's assertion that OFT is better than FOP is. It seems likely, though.

Integrating a winning percentage calculation into simulation is actually easy to do. There are three components necessary.

First, you need a winning percentage evaluator. This is simpler than searching to the end of the game. Fortunately, MAVEN already has a winning percentage evaluator, so it simply needs to be called within the simulator.

Second, the simulator must be modified to return both a score and a point differential. It might seem that you only need winning percentage, but you actually need both. There are situations where all of the moves win always, and then MAVEN should keep up appearances by selecting the move that maximizes point differential.

Finally, you need a policy for pruning and selecting moves that are evaluated using a two-dimensional evaluation function. On second thought, MAVEN's existing policies are good.

I actually implemented this capability while preparing this thesis, because the notes to the game Wapnick-Cappelletto (see section B.4) benefited from winning percentage analysis. I am still weighing evidence concerning whether using winning percentage results in better play than using point differential, so I will refrain from providing further detail.

11.4 Speed

Speed is less important than it was a few years ago, when sufficient speed was the difference between being able to do simulations and being unable to do them. Still, there are advantages to being fast.

First, you can consider more moves. You have already seen that there is a chance of missing the best play even if you consider 10 moves. It is helpful to systematically reduce the chance of missing the best move by increasing the number of moves.

Second, you can simulate candidates for additional iterations. If you quadruple the number of iterations then you reduce the impact of statistical noise by a factor of two. Reducing noise by a factor of two has two advantages. First, it is less likely that an inferior move will be selected. Second, and less obvious but just as significant, is that those bad moves that are selected are necessarily closer to the best move, so less is lost per error.

Third, against human opponents you can decrease the time required for moving, which places time pressure on opponents. Scrabble is a timed game, in which each minute of overtime costs 10 points. To be objective, a 10-point penalty is not that big a deal, since 10 points decide only a few percent of games. Perhaps panicking over the possibility of a 10-point penalty is actually more damaging. Against MAVEN, humans are unlikely to be far in the lead at the end of the game. MAVEN has an average spread of about +60 points in its tournament and match games, so every 10 points that a human loses puts his chances in grave jeopardy. What's more, even if the opponent does not go over, and even

if he does not panic, there is still a benefit to rapid play: the opponent's normal sense of the pace of the game is disrupted. You really have to watch humans to understand the impact of quick play. Humans are naturally jumpy because of the competitive situation, and additionally tense because they know that MAVEN is a strong player, and on top of that, it makes its moves so fast that their normal thinking process is disrupted.

Of course, the easiest way to make simulation faster is to wait for Intel to make faster processors. By consistently applying this technique for the last 5 years, MAVEN's simulation speed has improved by about a factor of 6.

If MAVEN is serious about speed, then it should try Gordon's move generator. Gordon measured [22] a factor of two improvement in speed from the GADDAG data structure, which is consistent with data from Appel and Jacobson's original paper [21].

The next easiest technique is to build a parallel simulator. Simulation is easily parallelizable, and interesting technical questions are involved. There is a simulator named PIOBOTJR that employs multiple processors.

Parallelization has a few interesting questions. A serial simulation controller can assume that it has complete information about all prior trials. That would be invalid in a parallel controller. How does this affect the controller? How large a speedup can you get on a parallel computer? I conjecture that parallelism will be close to perfect.

If speed is so important that you want to work hard, then you can tackle the question of fine-grained parallelism. For instance, if you have 10 moves in the simulation then on the opponent's first turn he will generate moves for 10 boards that differ by at most two moves. Can you exploit the similarity of the boards to generate moves more quickly? A 30% speedup may be possible from the reduction in move generation costs alone. If there is a way to share other overhead (e.g., precalculation of rack leave values) then the speedup could be larger.

Another idea is Graham Toal's Global Analysis. The idea is similar to how MAVEN handles endgames: throw all of the unseen tiles into one mega-rack, then generate all of the moves. The resulting list of moves contains every move that could be played. Sort the list into decreasing order, and index by the tiles used. With such a data structure, it is easy to compute the ideal move from any set of tiles with a few lookup operations. Any position that occurs often enough in a simulation should have a Global Analysis, so that move generation costs for that position will be essentially zero.

My perspective on fine-grained parallelism is that a speedup of perhaps 50% is unimportant given that Intel will handle it before the end of the year. And the same for Global Analysis.

11.5 To Challenge or Not to Challenge, That is the Question

MAVEN's current behavior is to automatically challenge phony words. This is usually correct, but can we improve upon that behavior?

The most frequent case of allowing a phony to stand is when you have a big play off it. Another case is when allowing the phony to stand reduces the game to a guaranteed win,

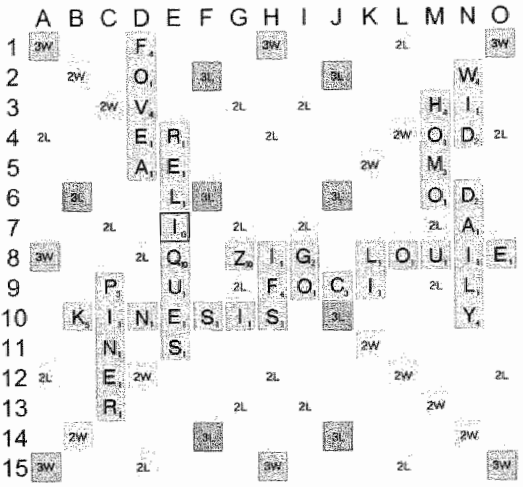
whereas challenging might allow the opponent to get back in. That requires understanding winning percentage.

A story is in order. Robert Felt described a game he had just played. His opponent played a phony bingo. Felt was about to challenge it off, but then he thought further. He calculated his best move assuming that he let the phony stand, and it turned out he had a bingo. If he challenged, he calculated that he had a different bingo. So far, the calculations indicated that a challenge was in order. However, upon closer inspection, it turns out that if he challenged and then played his bingo, the opponent would have a legal triple-triple. Felt decided that it was better to let the phony stand, since the exchange of bingos would ice the game. Felt was pleased to have found a game that MAVEN would have lost by automatically challenging.

Position 11-4 is from Wapnick's book [8] where an expert (Steven Polatnick) made the wrong decision. To quote Joel Wapnick,

> "After I played WID, Steven called "hold." He knew that WID was not good, and he was checking to see if it was to his advantage to leave it on the board. He allowed it to stay, and he was wrong to do so."

Polatnick played JETE (O1, 52, EEN), for a net gain of 30 points. But he should have challenged and then played JET (L4, 46, EEEN) for a net gain of 46 point. On the rack leave side, Polatnick has to accept a tripled E, but returning WID to Wapnick's rack is full compensation. Failing to challenge seems to be a 16-point error.



Position 11-4 A Challenging Position

Wapnick speculates [8] that Polatnick may have been concerned about giving Wapnick a play down the N-column, with the W on N4. That may be a factor, but if so then it was another error—if Wapnick had such a play he would have played it instead of WID, After all, the W plays on N4 whether the J is on L4 or not, and would score more than WID. Is it possible that Polatnick overlooked JET (L4)? There are a few decent J plays (JEE (F2, 29, EENT), JET (L2, 29, EEEN), and JEE (N2, 28, EENT)), but nothing that makes challenging obvious. JET (L4) is an obvious play, and Polatnick should see JET, but the best explanation is that he missed it.

In general, seeing these cases requires simulations, preferably conducted after inferring what tiles the opponent could be holding that could motivate a play like WID. Incidentally, inferences after a phony move can be difficult, since the engine might infer that *any* move would have been better than WID.

11.6 Specifically Defeating Weaker Opponents

How to play to defeat humans by exploiting their weakness is an open question. At various points in this thesis, the author has considered the impact of MAVEN's style of play on human opponents. The following list summarizes a few of these.

1) After discovering that there is a theoretical bias in favor of short plays on the first turn, the author considered whether that was good strategy for MAVEN. MAVEN benefits from having a wide-open board, as even the best humans miss 10% of their bingos, whereas MAVEN never misses.

2) MAVEN's simulator gives the opponent additional thinking time, which is a significant practical drawback because it takes time pressure off the opponent.

3) MAVEN's endgame player foresees defenses that the opponent is likely to miss, in the process foregoing a chance to run up a big score at a small risk.

4) MAVEN's pre-endgame evaluator decreases MAVEN's chance of mounting a comeback, but also decreases MAVEN's chance of being overtaken. The author decided that tradeoff was a wash against other computers, but an advantage against humans.

5) In considering MAVEN's provocative style of wide-open games that completely disregard conventional defensive precepts, I decided that it was best if MAVEN's opponents held contempt for its positional skill. Many years passed before humans realized that MAVEN was correct.

In each case, MAVEN went with the theoretical predictions rather than warp the program to specifically defeat humans. However, at this point we can regard the problem of choosing the theoretically best plays as essentially solved. Therefore, the question of whether risks are justified in the pursuit of victory is one of the biggest open questions.

There is a practical motivation for pursuing such a line of research. Studies have shown that weaker players defeat stronger players more often than would be predicted by the difference in rating [50]. This is definitely true of MAVEN; there is an 1800 rated player who can defeat MAVEN (non-simulator) about 25% of the time. The difference in rating (300 points) suggests that she should win only 16% of the time. Her experience with MAVEN confirms the author's. In addition, a multiyear study of all tournament games shows that the same is true of human-human encounters as well.

It is an open question why this is true. The rating system is essentially the same as used in chess. We can rule out the possibility that the rating results are incorrect, since many people have checked their values. The author speculates that the variance in a game is larger when strong plays weak, but there is no data to test this hypothesis. Another possibility is that there is a relatively constant chance that the opponent will draw both blanks, and then he has a relatively constant chance of winning.

At any rate, if you want optimal results, it is important to beat the daylights out of weaker players. Championships in Scrabble are not decided by head-to-head matches between

top-ranked players[47]. Instead, the Champion is the player with the highest score in a tournament in which the average participant is much weaker than the eventual winner is. For instance, the North American Championship is a 31-round event in which the average participant has about a 1900 rating. The best players in the tournament have ratings around 2050, so they should win 67% of their games. The tournament winner has averaged 80% wins.

The goal of exploiting weakness is not to beat the strongest players more often, although that may happen as a side effect. The goal is to defeat players rated about 200 points lower. You should defeat them 77% of the time, but in practice, you fall short, winning perhaps 67%.

We have already considered that modeling the opponent's weakness would produce moves that tend to win more often. We can consider such a solution as a righteous approach to the problem. In this section, we will cover a more devious approach: playing phonies and swindles.

11.6.1 Phonies

Humans below the top class do not know the words solidly, especially the lower frequency words. Thus, it is possible to get away with phony words. But significant challenges (pun there) lie ahead.

First, there is the question of what makes a plausible phony. The answer is a complicated function. For example, from a morphological perspective the word EULOGIAE is highly implausible. Few words contain letters in such combinations. Nevertheless, the word is good, and known to all experts. By contrast, BIRTHDATE is phony and no human would give it a second thought.

Here is how evaluating the plausibility of phonies might proceed. You can determine whether a word would land on lists of high frequency words such as the 5-vowelled-8's or JQXZ words. If a phony ought to be on such a list then it is immediately an implausible phony because a human would realize that he has not seen it before. Another dimension is the frequency of the word. Word frequency can be measured for real words by playing many games and recording the frequency of words as a function of their length and letter content. You can project phonies onto the same scale. Playing a phony that should appear often differs from playing a phony that should rarely appear.

Another factor is the strength of the opponent. Stronger players know more words, so plausibility should be a function of strength.

Certain phonies are automatically implausible. If the opponent is a computer then the phony is impossible. If the word occurred in one of the opponent's games then it is implausible. If the opponent challenged it then it is highly implausible. If the program's previous move was successfully challenged, then a phony on this turn is implausible. If the phony is the last move of a game then it is implausible, since the opponent risks nothing by challenging.

---

[47] Actually, the World Championship employs a best-of-5 match. But the point stands because to qualify for the final match you have to be one of the top two finishers of a Swiss tournament.

Once you have a reasonable evaluation function for plausibility, the next task is to create a list of phony words. You have several sources. For example, you can spy on Internet games and await phony words from the players. You can spell-check Web pages from sources with good editorial controls (e.g., Yahoo). Humans have collected lists from tournament play. One fellow collects challenge slips from tournament word judges. In addition, you can make them up yourself by writing little programs that compose plausible variations on acceptable words, such as adding prefixes and suffixes. Of course, you should check every word because some will be outlandish. You should build the list from many sources, because you need many words (tens of thousands) before they have an impact on winning chances.

If you ever generate a phony play, then you must weigh phony plays against real plays. That is a complicated problem. From the opponent's perspective, a phony word is simply a word he has never seen before. These come up from time to time, since the opponent does not know all of the words. The question from your opponent's perspective is to judge the likelihood that an unfamiliar word with that morphology is phony. The opponent has two frames of reference for answering that question.

First, he can judge from his overall experience with words of that type. The goal of rating a word for plausibility is to provide MAVEN with the opponent's judgment from this frame of reference. That is, the plausibility of a word is defined to be the likelihood that an unfamiliar word with a given morphology is acceptable.

Second, he can judge based upon MAVEN's perceived tendencies. That is, he can consider MAVEN's history of playing words with the given morphology and count how many phonies he has seen. Where MAVEN has many phonies with a given morphology and few real words then MAVEN has to throttle the phonies such that they occur at a rate that does not arouse suspicion.

If you do all of these things then you have a model of how the human rates the probability of the word being phony. If he is rational then he will challenge if doing so increases his winning percentage, and he will not challenge otherwise. So you need to model how a player of a given skill level perceives his winning chances against MAVEN as a function of the score and the stage of the game. You could start by doing a theoretical model, or you could capture real data by spying on Internet games between humans.

These calculations allow the program to decide whether to play a phony: just compute the winning percentages to determine whether the opponent should challenge. If he should challenge, then do not phony. Otherwise, wing it.

The model just described places itself in the opponent's position and tries to weigh the odds. A different point of view is to regard the opponent's challenge as a random event that the program must strategize using probability-weighted search. For example, if you have enough Internet data then you can train a neural network to predict the frequency of challenges as a function of the player's rating, the word's plausibility, and the true winning percentage. The program will decide to play a phony if its winning percentage is greater when weighted by the chance that the opponent challenges.

You may need a model of the "advertising" benefit of playing phonies. One of the advantages of playing phonies is that the opponent will sometimes challenge good moves. This never happens to MAVEN, since opponents know that MAVEN never plays phonies. How do you account for this?

Playing phonies is a large research project. You will need a large knowledge base of phony words. You will have to grade the words for plausibility. You must tune the psychological model of the opponent's reaction to unfamiliar words. Constant maintenance will be required. Tough job.

11.6.2 Swindles

Simulating games using a suitable opponent model would generate many moves that qualify as "playing to the opponent's weakness." This section will focus on true swindles. A swindle is a play that should fail against perfect opposition, but the player has determined that against his actual opponent there is a lot to gain and little to lose. Master players take the greatest pride in playing swindles, because such plays demonstrate both technical and psychological mastery.

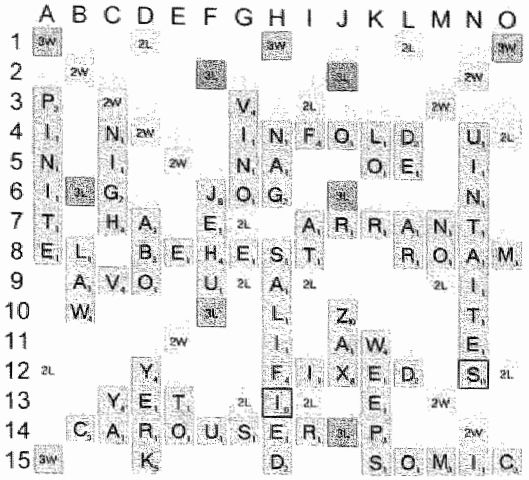
The gain versus loss balance for some swindles is easy to measure. For example, in an endgame proper you can determine the true score after the perfect play, and the true score after the swindle. The advantage is what you stand to lose by playing the swindle. It would be grossly negligent to attempt a swindle if the potential loss puts winning in jeopardy, so we can restrict attention to cases where only points are risked, including cases where the opponent can lose if he goes astray.

The potential gain of an endgame swindle is harder to judge. You cannot know which move the opponent will play. The opponent might find the ideal move, or he might find several potential alternatives. To evaluate a swindle you must guess the odds of each error.

Robert Watson reached Position 11-5 against Peter Morris in a game annotated in *Scrabble Players News* [66]. Watson is moving.

This is an out-in-two position, decidedly not a one-tile position, because Morris ought to play out quickly; Watson's consonants would badly beat Morris's vowels in a one-tile game. Therefore, Watson should size up Morris's out-in-two threats. The highest is OUT (D3, 12) combined with VEE (C9, 11). Note that

1) VEE blocks DRY (12B, 13).
2) If OUT is not out, Watson has GLOUT (D1, 8) off it.
3) If VEE is blocked, then Morris has no good plays for EE.
4) OUT does not block any of Watson's good scores.



Position 11-5 Swindle Example

Thus, Watson should play DRY immediately. However, Watson played NOB (M7, 15), thinking that it is a one-tile situation. Analysis shows that NOB is a four-point error. The variations are given in Table 11-3.

Note, however, that once a player starts off with the wrong idea, he is unlikely to immediately realize that he should change plans. If Watson believes that he should play one tile at a time, then he might not follow up correctly after playing NOB. Errors that originate in a positional or strategic misunderstanding tend to cascade.

Word	Spot	Score	Best	Spot	Score
DRY	12B	+13	NOB	M7	+15
ETUI	11E	-8	VEE	C9	-11
NOB	M7	+15	JEHUS	F6	+17
OE	11A	-11	OUT	D3	-12
(GLQS)		-28 = -19	(DGLQR)		-32 = -23

Table 11-3 Best Play, and Best Play after NOB

Morris followed NOB with OUT (6J), which created the phony DETAR. Watson challenged, and the game was routinely played out, as shown in Table 11-4.

Player	Word	Spot	Score	Best Play
Watson	NOB	M7	+15	DRY (12B, 13)
Morris	OUT	6J	0	VEE (C9, 11)
Watson	JEHUS	F10	+17	DRY again
Morris	OUT	D3	-12	VEE again
Watson	DRY	12B	+13	Best
Morris	AE	9H	-7	Best
Watson	GOUT	D2	+5	GLOUT (D1, 8)
Morris	EF	I3	-6	LAWED (B8, 9)
	(QL)		-22 = 3	

Table 11-4 Game Continuation

Morris's play of OUT requires examination. Morris says that the following line of reasoning lead him to play OUT:

1) Watson can only lose by challenging a phony word.
2) Watson is probably not sure if DETAR is good.
3) So Watson will not challenge.
4) Therefore, OUT steals a few points for free.

This section will examine this reasoning from the perspective of an endgame swindle. What is the risk/reward ratio of the phony OUT?

Joe Edley, who annotated this game, restricted himself to a comment that Morris's OUT (6J) was "a last effort to close the gap." However, the gap is too large even if Watson fails to challenge *and* plays inferior defense. Table 11-5 shows two ways to win for Watson, who is ahead by 37 points. This establishes that winning and losing are not at stake. It follows that we will not consider extreme sacrifices of points in order to win.

Word	Spot	Score
NOB	M7	+15
OUT	6J	-19
ODORS	J4	+22 (!)
VEE	C9	-11
(GLQR)		-28 = -21

Word	Spot	Score
NOB	M7	+15
OUT	6J	-19
DRY	12B	+13
NEE	4C	-6
(GLQS)		-28 = -25

Table 11-5 OUT (6J) Loses Even if No Challenge

This idea is analogous to a bluff at poker. We can assess the risk/reward ratio for this endgame bluff by calculating the best-play sequences after Morris plays his best move, VEE (which nets –23 as shown in Table 11-3), and after OUT if Watson challenges and if Watson does not challenge. The leftmost variation of Table 11-6 shows that if Watson fails to challenge DETAR then Watson nets -21. Thus, Morris stands to gain 2 points if his phony stands. The following variation shows that if Watson challenges then Watson nets -4:

Word	Spot	Score
NOB	M7	+15
OUT	6J	0
DRY	12B	+13
OUT	D3	-12
JEHUS	F10	+17
AE	9H	-7
GLOUT	D1	+8
LAWED	B8	-9
(Q)		-20 = -4

Table 11-6 Best Play after OUT is Challenged

Since Watson would gain -23 after Morris's VEE, this variation shows that OUT is, in perfect play, a 19-point error. We can now assess Morris's bluff: Morris risked 19 points for a chance at 2.

If the game were on the line then the phony would have something extra going for it. However, as the score stands, risking 19 points to gain 2 is obviously wrong. Morris misjudged the situation.

Pre-endgame swindles are murkier. Most pre-endgame swindles are fishes for monster plays. Often these are justified without reference to swindling chances, since the opponent need not make an error for the plan to succeed. However, it would qualify as a swindle to fish for a blockable bingo because you need your opponent to err. Possibly, we can generate and evaluate such plays.

11.6.3 Integration

The design above counts on simulation being a "grand unifying theory" to pull these elements together. To pull it off, you would need a model of a player of a given rating. The model must exhibit the exploitable behaviors that characterize players like the opponent. If the model misses 8-letter bingos at the proper rate, then the simulation will create opportunities for the opponent to miss them.

One complexity is that the endgame model must generalize to integrate the opponent model, since we no longer postulate deterministic play by the opponent.

Chapter 12 – Competitive Results

MAVEN has always focused on defeating the best human players, so its competitive results are worth evaluating.

The first section describes six metrics that we can use to evaluate programs. The metrics differ in the qualities that they attempt to measure, so it is best to use all of them to get a true picture of the program. The second section gives the competitive results of all of MAVEN's tournaments and matches. The final section summarizes MAVEN's Man-Machine encounters and asserts that MAVEN is of superhuman strength.

12.1 Standards of Evaluation

We can evaluate MAVEN's performance using many metrics. This section describes some metrics that have been applied to MAVEN.

12.1.1 Winning Percentage

The obvious measure of skill is winning percentage. Alas, this measure suffers from a key drawback: the quality of one's opponent matters a great deal. If we consider a "level" to consist of a difference of skill that corresponds to a 75% winning percentage, then the ranks of human tournament players contain at least six levels. Therefore, we must qualify any winning percentage measurement with a "strength of schedule" adjustment.

12.1.2 Rating

Tournament players use an Elo rating system [67] to grade players. Both competitive results (wins, losses and draws) and the strength of the opponent determine how your rating changes. On that scale, novices rate around 700 and the highest player is usually around 2100. A 200-point differential in rating corresponds to a 77-percent winning percentage. Any player rated over 2000 has a legitimate chance to win the North American Championship.

Rating is a good measure of skill, but it has difficulties. First, rating is not a stationary function; the skill of a player varies over time. To maintain a level of skill requires intensive training, and it is inevitable that players will take extensive breaks from training from time to time. Nearly all of the top players have taken time away from the game because of burnout. Even if players do not take time away from the game, it is difficult to maintain competitive drive year after year. Therefore, players have rating fluctuations that are too large to explain by relying on chance alone.

Ratings also reflect the inherent variability of winning percentage. Even when a player's skill is not fluctuating, his rating will fluctuate, usually by 30 to 50 points around his true skill level. For example, the winner of the Nationals might score 25-6 over a field of 1950 strength. This performance would be characteristic of a player having a rating of around 2300, so the winner's rating rises by a lot, often to well over 2100. However, is he really better than he was before the tournament, when his rating was 2050? After all, *somebody* had to win the event.

One technical problem with rating is that the NSA rating system has drifted over time. Because rating systems rely only on the difference between player ratings, there is no rating that represents a fixed level of skill. This has been a problem for MAVEN, since MAVEN has competitive results over a 12-year period. During that period, the ratings of the top players have drifted downwards by perhaps 50 to 75 points. In this thesis I have attempted to correct for such drift by increasing the ratings of MAVEN's opponents in 1997 and 1998 to the level they would have had in 1986. This adjustment allows us to compare MAVEN against the historical records of Ron Tiekert (2170 peak) and Robert Felt (2140 peak), for example. Still, this procedure is open to obvious objections, and will not stand up over time.

The real problem is that MAVEN has not competed often enough to acquire a reliable NSA rating in any single time period. It has been difficult to find opponents.

12.1.3 Scoring

One of the reasons that winning percentage is a difficult measure to work with is that it discards information. If we know that a final score was 499-200 then we have a lot more information than the 1-0 winning percentage suggests. For instance, although MAVEN and Adam Logan split the first six games of their 1998 match, MAVEN's point differential was far higher. It was clear that MAVEN had simply been unlucky to lose a few close games, and if that luck evened out then MAVEN would win the match comfortably.

The obvious measure of skill that considers score is the average number of points scored per game. Obviously, if Player A averages 450 points per game and Player B averages 350 then A is a better player than B. However, this metric has a defect, and if you play Scrabble for any length of time then you will find this out. The quality of the opponent matters a great deal; one can score more points against weaker opponents than against top masters. Players A and B could be roughly equal players, but A has played weaker opponents than B.

The reason why this happens is that the number of turns in a game increases when the players are weak. Strong players use 4.5 tiles per turn, so the games are finished in 11 to 12 turns per player. Weak players may only dispose of three tiles per turn, which extends the game for 4 more turns per player. In a matchup of strong versus weak players, the strong player has the chance to play more moves than he normally would when playing against his peers. He also wins challenges and gets away with phonies, which further increases his score.

12.1.4 Points Per Move

A slight modification to this metric, measuring the average number of points scored per turn, is a much more reliable indicator of a player's skill level. It works because both players have the same number of turns per game. Indeed, this measure is remarkably consistent.

An opponent's style can influence the number of points-per-turn scored. Some players create constricted positions in which it is difficult to score. Other players prefer wide-open positions. There are limits to the impact of this factor, since carrying stylistic preferences to an extreme will reduce your winning chances by more than it hurts the opponent. Experience suggests that if defensive measures reduce your opponent's score

by more than a couple of points per turn then you are hurting yourself more than your opponent.

Rating is a good measure for assessing tournament play, since it permits indirect comparisons of players by "factoring out" the quality of the opponent. However, rating is a more variable function than points-per-turn. MAVEN's self-play games contain 21 moves on average (almost 11 per player). So estimates based upon points per turn converge $\sqrt{21}$ times faster than estimates based upon rating.

12.1.5 Scoring Statistics

MAVEN averages 35.0 points per move. MAVEN-versus-MAVEN games are over in 10.5 moves per side on average. Each MAVEN plays 1.9 bingos per game. These statistics are considerably higher than human expert statistics. The best humans average around 33 points per move, 11.5 moves per game and about 1.7 bingos per game.

12.1.6 Test Beds

During development one often needs to experiment with some aspect of the program while ensuring that other aspects remain stable. For instance, one may wish to develop new endgame evaluation functions, while ensuring that MAVEN's endgame skill level remains constant. Test beds consisting of validated positional examples are useful here. In fact, it would be very dangerous to make changes to MAVEN without this basic safeguard. MAVEN is already a highly tuned system, so changes are likely to be for the worse unless proven otherwise.

Sadly, there is no public test bed, such as the Bratko-Kopec test bed for computer chess [58]. In my development work, I usually construct test beds for specific features as needed.

12.1.7 Choice of Metric

We do not really need one single measure of strength; the important thing is to have measures that are appropriate for what we are trying to accomplish. In comparing consecutive versions of MAVEN, points-per-turn is useful since it is easier to reach conclusions quickly. In comparing MAVEN with humans, rating proves to be the most useful metric.

12.2 Tournaments and Matches

12.2.1 First Impressions

We have already described MAVEN's first tournament, in December of 1986. The caliber of the opposition was very high. MAVEN made a big impression, scoring 8-2 with a +70 point per game differential.

12.2.2 Second Look

MAVEN's second tournament was in October 1987. The Cape Cod Fun Weekend was probably not a good event for a computer to enter. Several of the contestants were upset, despite our contribution of a prize to the best game against MAVEN. The opponents were

experts, but below championship caliber. MAVEN won 5-0 with an average margin of victory of over 70 points.

12.2.3 Close Encounters of the Third Kind

The December 1988 tournament was a "team tournament" in which a computer team was entered. If we had entered a team consisting of the best available programs of the day then we would have won the event handily, because TYLER went 6-4 on first board, MAVEN went 7-3 on second board, and programs of the strength of MAVEN and TYLER would have cleaned up on third and fourth boards. However, dreadful commercial programs filled out the team. The program on fourth board lost every game by using too much time.

Still, it was a good event for MAVEN. MAVEN's tiles were not good overall, as it drew only 8 out of 20 blanks despite playing 55% of the tiles, but it still performed impressively. MAVEN's point differential exceeded 70 points for the third event in a row. MAVEN's first successful endgame player made its debut in this event, improving on MAVEN's play in several games. MAVEN also contributed insights to the postmortem analysis of other games. However, on this last subject we are of two minds; is the automation of kibitzing really a good thing?

Interesting: MAVEN lost its game against the weakest opponent it has ever played. Mark Berg (NSA 1600) defeated MAVEN. Berg drew both blanks on his first turn, and played TRODDEN, the only bingo of the game. MAVEN drew the Q twice and had to exchange it twice. Then MAVEN drew the Q at the end of the game and could not play it. Despite all of this good fortune, Berg managed to win by only a few points. This shows how much has to go wrong for MAVEN to lose to a 1600-rated player.

Alas, MAVEN has had no further tournament games. The consensus within the Scrabble community is that tournaments are for people only.

12.2.4 Informal Data

A few years ago, a demo version of MAVEN circulated within the Scrabble community. This engine contained only the Basic evaluator, the pre-endgame evaluator, and the endgame search engine. In particular, the program could not choose its moves using simulation.

Numerous top players (including North American and World Champions such as Joel Wapnick, Robert Felt and Peter Morris) used this program for training. No player reported being able to win more than 48 percent of his games. Given that MAVEN has made significant advances in the interim, the gap between MAVEN and the top human players has grown substantially.

An important believer in MAVEN's analysis is Joe Edley, the editor of the US National Scrabble Association's newspaper. Edley decreed that the NSA would only publish annotations of games that MAVEN simulations had verified

12.2.5 Hall of Champions, 1997

Matt Ginsberg organized the first AAAI match for the Hall of Champions demonstration. MAVEN was in dreadful condition for the event. The author did not leave sufficient time to prepare, and discovered at the last minute that a compiler "upgrade" was incompatible with the code. It required consecutive overnight efforts just to make something that could play at all. The user interface was dreadful and error-prone. MAVEN had avoided this pitfall in other events, in accordance with the theory that it is more important to deploy sound software than cutting-edge features.

MAVEN deployed a simulation capability for the first time in the AAAI-97 match. The software was not ready. MAVEN's development computer hardware was much slower than the target hardware, so the author had to guess how many moves to simulate for how long. Both guesses were inappropriate. There was no time to work out how to apply simulations to pre-endgames, so MAVEN played pre-endgames using its pre-endgame analyzer. The result was an ugly 0-2 loss to Adam Logan. In truth, MAVEN did not play that badly, but the result reflects badly on the author.

12.2.6 New York Times Match

A reporter for the New York Times named John Tierney compared the world's best human players with MAVEN. He arranged a best-of-11 match between MAVEN and World Champion Joel Sherman[48] and World Runner-up Matt Graham. Sherman and Graham competed as a team against MAVEN.

MAVEN did not employ simulations for the match, so the players were competing against the basic MAVEN AI, which selected moves using the heuristic rack evaluator in the early game, the probability weighted search engine in the pre-endgame, and the B* endgame search engine.

MAVEN won by 6-3. One cute play: MAVEN held AIIRMSU and found an open T for TIRAMISU. For a human this is a difficult find, because the word has neither prefixes nor suffixes, and there are a number of prefixes and suffixes in the rack (AIR, ISM, MIS) to distract attention. The word itself has an S in seventh position, and a U in eighth, which are both rare.[49]

12.2.7 Hall of Champions, 1998

Jonathan Schaeffer and the author were disappointed in the outcome of the first Hall of Champions match, so we arranged another. (One of the advantages of being the challenger is that you can keep playing until you win.) Adam Logan graciously agreed to a rematch. MAVEN spent four solid months "training" for the event, and MAVEN was in good condition in most respects.

---

[48] Interesting coincidence: because of the match I discovered that Joel Sherman and the author went to the *same high school*. Moreover, we are the *same age*. This sent me scurrying to my high school yearbook. It turns out that we are classmates, and because Sheppard and Sherman are adjacent alphabetically, our pictures are adjacent in the yearbook. What are the odds?

[49] At dinner that night, Joel and Matt ordered tiramisu for dessert.

Not wanting to leave matters to chance, we made the match significantly longer. Jonathan asked, "How many games do you need to prove superiority?" An good question, as most people underestimate the necessary number of games. MAVEN should win about 60% against Logan, and you would think that such an advantage must surely prove decisive in a series of 14 games. Actually, the chance of winning at least eight games out of 14 is only 69.25%. MAVEN would actually be better off with only a 13 game match (winning chance 77.12%) because a 7-7 standoff is impossible. However, in Scrabble you must alternate who moves first, so an even number of games is forced.

It is not until you play 20 games that a 60% edge results in a match victory 75% of the time. However, asking Logan to play 20 games in two days is grossly unfair. Playing 14 games in two days is normal, however. For example, in the North American Championship tournament, the players play seven games for four consecutive days, followed by three games on the fifth day. Playing a 14 game match was a fair test by the standards of human master practice.

MAVEN won by 9-5, and analysis of the moves shows that MAVEN played better. The luck of the draw was equal in most respects. A later section will summarize the moves of the match.

12.3 Man-Machine Competition

Table 12-1 shows MAVEN's public tournament and match results since its inception. We estimate the strength of human champions at 2050, rather than the inflated post-championship ratings of those players. Adam Logan, for example, had a 2125 rating when MAVEN played him, but it is absurd to regard that as a realistic representation of Logan's strength, since the ratings of champions always decline after their luck runs out. The choice of 2050 seems like a reasonable estimate for a human champion, particularly because the table mixes rating data over a 12-year period.

Date	Event	Opposition	Rating	Result
December, 1986	Matchups Tournament	Top experts	1950	8-2
October, 1987	Cape Code Fun Weekend	Pairs of experts	1850	5-0
December, 1988	Matchups Team Tournament	Wide range of experts	1875	7-3
July, 1997	Hall of Champions 1997	Logan	2050	0-2
March, 1998	New York Times Match	Sherman and Graham	2050	6-3
July, 1998	Hall of Champions 1998	Logan	2050	9-5
1986-1998	**Total**		**1975**	**35-15**

Table 12-1 Man-MAVEN Competitive Summary

MAVEN's total match and tournament record is 35 wins and 15 losses against an average rating of 1975. A 70 percent score against such opposition is a major success. MAVEN's performance rating for these 50 games is 2125. I estimate that the simulation player would earn a rating of about 2150.

While the *highest* rating ever achieved by a human is 2170, no player has maintained a rating over 2100 for two consecutive years. The pattern is for the rating of the NSC winner to shoot above 2100, where it remains until he plays enough games to deflate the rating. Depending on how often that player competes, it may take a year to come down to earth, but decline is inevitable. No human has ever maintained a rating over 2100 for long.

12.4 Analysis of Errors

This subsection tabulates errors committed in the 1998 AAAI match against Adam Logan, which MAVEN won by 9 to 5. Appendix B analyzes the moves of the ninth game in that match. Logan was (and still is) one of the top six players in the world. He finished fourth in the 2000 World Scrabble championship.

The match contained 14 games encompassing 168 move decisions per side. Logan played well, but it is abundantly clear that MAVEN played better. MAVEN's errors came from three sources:

1) A bug that prevented it from choosing exchanging moves. There were three moves in 14 games where it should have exchanged but did not, costing 17 points.

2) A bug caused the endgame player to stop search early. An "optimization" inserted to speed pre-endgame search was incorrectly used in the endgame proper. Fortunately, it was impossible for this bug to affect the outcome of a game.

4) Search control issues. MAVEN cut off a move that proved to be best in a few instances, costing 7.2 points. Then there were cases in which MAVEN did not consider the best move. It is hard to estimate how many errors were attributable to missing the best play, since if MAVEN does not generate the move then it is unlikely that I find it either. Of course, MAVEN can simulate any such move; the only problem is to generate possibilities. Consequently, there is a novel way to measure the quality of the search controller. MAVEN annotated Logan's moves, and recorded how often Logan's move was better than any move that MAVEN generated. This analysis showed that Logan's move was better 7 times in the 14 games, yielding a total of 68.5 points. Of course, Logan might not find the best move every time, so this is a lower bound on MAVEN's error rate.[50] A likely upper bound is 14 errors in 14 games yielding a total of 137 points (9.8 points per game), on the assumption that Logan will come up with the best move on at least 50 percent of his plays.

[50] However, more recent simulations of MAVEN's moves using simulations that included up to 25 moves failed to identify any further errors. It is the author's opinion that there are none.

Type of Error	Instances	Points Lost
Failed to exchange	3	17.0
Endgame search truncated because of buggy optimization	4	22.0
Move not included in simulation	9	80.4
Move inadvertently pruned	3	7.2
Need to understand winning percentage	3	2.4
Total	**22**	**119.0**

Table 12-2 Error Analysis for MAVEN (28 Games)

The post-mortem analysis (Table 12-2) used MAVEN to annotate all moves made by either side, so considerably more than 14 games are involved. All told, there are 357 move decisions in the sample, so the error rate is exactly 1/3 point per move. In a typical game MAVEN will have about 12 move decisions, so the error rate is about 4 points per game. In truth, this is a lower bound on MAVEN's error rate. As described in point 3, above, there could be 7 more errors totaling 68.5 points that we have not discovered. Adding 68.5 points to the total produces 187.5 points of errors, which is about 6 points per game.

Analysis of tile distribution shows that luck divided equally, at least in gross terms. Blanks, Q's, S's, and so on were well within the norms for a short match. Each side had one last-gasp victory.

Type of Error	Instances	Points Lost
Missed bingos	3	85.1
Minor inaccuracy	27	39.6
Substantial errors	21	94.7
Pre-endgame errors	17	141.6
Endgame errors	15	171.0
Total	**83**	**532.0**

Table 12-3 Error Analysis for Adam Logan (14 Games)

Table 12-3 summarizes Logan's errors. There are a few points to note about Logan's error distribution. First is that almost 60% of the points were lost in the pre-endgame or endgame stages. Though these make up less than half of the plays, it is easy to make a large slip in those stages. The early game had more errors, yet fewer points were lost.

Logan's errors total about 38 points per game, versus MAVEN's 6. The match outcome of a 9-5 victory for MAVEN is a reasonable reflection of Logan's disadvantage in point differential. Incidentally, a non-simulator MAVEN was tested on these same 14 games, and it averaged about 29 points of errors per game. This confirms my assertion that non-simulator MAVEN has an edge over the best human players, albeit a small one.

Chapter 13 – Frontiers

The biggest challenge in Scrabble AI is met, because no one seriously disputes that MAVEN is better than all humans. Strictly speaking, this has been demonstrated only within North America. A different dictionary is used in the rest of the English-speaking world, and there are other languages. However, adapting MAVEN to a new dictionary is not a challenge. This has already been done for TWL98, SOWPODS, French, Dutch, and German. There is no doubt in the author's mind that MAVEN is superhuman in every language.

The real challenge is to improve MAVEN, preferably to achieve asymptotic practical perfection. There are several avenues to explore, but the total improvement will be only a few points per game. This chapter summarizes the opportunities, and points the reader to other sections where additional detail is given.

13.1 Simulation Controller

It would be an improvement to make the simulation controller scalable, so that it operates on slower machines, and exploits faster machines. The actual cutoffs (e.g., 10 moves, 17 iterations before any pruning, 1000 iterations, etc.) were tuned by experience on a Pentium II/300. The author now believes that an ideal simulation controller would be able to run with *some* benefit even on a 486/66. Chapter 10 gives further details.

While writing the notes to the games of Appendix B, I created a simulation controller that appears to be scalable. I am not prepared to publish the results now, because I have not qualified the range of hardware over which it scales. But I have concluded that a scalable controller is feasible. Moreover, my implementation performs about the same as my hand-tuned controller when operating on equivalent hardware.[51]

13.2 Inferences

It may improve MAVEN to eliminate a modeling fallacy: MAVEN assumes the opponent's rack is drawn from a uniform tile distribution. However, we can infer a more accurate distribution of tiles from the opponent's last play. For instance, suppose the opponent's opening play is BARD (8G,14). Did he keep an O? No, certainly not, because then he would have played BOARD (8D,22). Did he keep an E? (No! BARED.) Did he keep an I? (No! BRAID.) Did he keep a U? (Probably not, since DRUB would be better than BARD, but he would play BARD if he held a duplicate A.) We can go on like this through all the tiles, and combinations of tiles, to come up with a list of racks that the opponent could have held. This process is called "inferring" the opponent's rack.

Inferences change the distribution of tiles that the opponent could have kept, which biases the future distribution. Refer to Chapter 10 for elaboration.

I have implemented an inference engine, because I wanted to determine whether the annotations of Appendix B were "stable" under inferred distributions. My finding is that the annotations are stable, but my implementation is not! I am reminded of the tweaking that the pre-endgame analyzer required. My inference engine is in its third revision, and

[51] Knowing this in 1998 would have saved me a month of tweaking.

still does not demonstrate clear superiority over using the uniform distribution. So for the purpose of this thesis I will simply state that it is a work in progress, and shows just as much potential as ever.[52]

13.3 Fishing

MAVEN's lack of a move generator for fishing may be its biggest weakness. A "fish" is a try at drawing the one or two tiles needed to convert the rack leave into a bingo. For example, racks like AEINST can produce a 7-letter bingo on 90 percent of the draws from the bag, so it can be lucrative to fish.

All of the moves that MAVEN overlooked in its match against Adam Logan match were fishes. In the days before simulation, it would have been pointless to generate fishing plays because the evaluation function would always place them behind the "normal" move. However, simulation rationalizes the comparison of moves that were generated by incomparable generators, allowing the power of good fishing moves to show.

Recent work on the scalable simulation controller shows that fishing plays tend to occur among moves 11 through 25 in the ordering of the Basic evaluator. It follows that deploying a scalable simulation controller will capture most of the benefit of fishing, without taking on the implementation hassle of developing a special-purpose evaluator. It remains to be seen if extreme fishes (e.g., exchange O out of AEIORST) are generated by a scalable controller.

Another possibility recently occurred to me. There are less than 4 million distinct racks of 7 or fewer tiles, so it is possible to learn a value for every single one. For example, we could initially evaluate the racks according to the Basic model, and then update the values based upon the results of simulations. This would give the rack evaluator better values, so it would play more strongly, and it would place strong fishing moves higher in the ranking. I particularly like this system because it is fully automated.

13.4 Winning Percentage

It will help to include a winning percentage estimator in MAVEN. MAVEN actually has such an estimator, using neural networks. I have recently deployed it during simulations, but it is not yet clear whether it makes a difference. Please see Chapter 6 for a more complete discussion.

13.5 Biases

After the simulation is complete, it often happens that two or more moves are so close that it really does not matter which is selected. Is there some practical benefit to one of the moves? For example, will the opponent make an error more often against one play than against another? The concept of an "opponent's model" is discussed in Chapter 10.

13.6 Challenge Tactics

The idea is to challenge when you gain more points by challenging, not simply because the opponent played a phony. For details and examples, see Chapter 10.

---

[52] Unrealized potential, that is.

13.7 Opening Move Evaluation

Steven Gordon has proposed creating a database of all possible opening racks together with their recommended moves under simulation. There are 3,199,724 racks of 7 tiles using the full bag [59].

This proposal may be impractical. Conventional evaluation functions often omit crucial fishing plays, so the proposal does not address the real problem. Even if that problem were solved, it would take many CPU years to complete the simulations of 3,199,724 racks. Plus, it would SUQ if the dictionary committee met on the QT over some slices of ZA with anchovies and added new words to the dictionary, thereby rendering the whole computation obsolete.[53]

Instead, a simple evaluator should be able to improve upon the evaluation of opening turns. The author would focus efforts there, and trust simulation to work out the details on the fly. After all, this policy will require less CPU time until you play 3,199,724 games.

Perhaps the benefit of precomputing the opening racks would be to enable accurate inferences on replies to the opening. For instance, for which opening racks would the opponent play PAY (8F)? Why did he not play YAP or PYA? Why was the play at 8F rather than 8G or 8H? All of these questions could be answered by an exhaustive opening library.

[53] SUQ was added in the 1993 revision. QT almost made the 1993 revision, but was omitted on the grounds that it was an abbreviation for "quiet." The word ZA, collegiate slang for "pizza," will probably go into the next revision.

Chapter 14 – Research Results

In Chapter 2, we asked certain questions that this research has answered, and it is worth specifically going over the answers in conclusion.

14.1 Competitive Demonstration of Superiority over Humans

Is it possible to build a computer program that outplays all human Scrabble experts? The answer is, "Yes, without a doubt." MAVEN outplays all human experts.

The first version of MAVEN overcame the three most obvious roadblocks that stymied previous Scrabble programs. First, MAVEN was fast enough to complete a game in tournament time. Actually, it was fast enough to put human opponents into extreme time pressure. Second, MAVEN had the complete list of legal words. Well, to be truthful there were some errors, but the vocabulary was not an obstacle to superb play. Third, MAVEN developed a theory of positional evaluation that was well founded. So well founded that human experts were at a strategic disadvantage until they adopted MAVEN's theories for themselves.

Another question was whether anti-computer strategies could specifically defeat MAVEN. The answer was "Yes and no." At first there were, as perceptive players could trick MAVEN in the endgame and pre-endgame. However, it must be noted that despite the possibility of pinning a defeat on MAVEN via late-game tactics, usually MAVEN was ahead by so much that there was no possibility of losing anyway. Few losses were ever inflicted on MAVEN in competitive results by any means whatsoever. In any event, such weaknesses only lasted for a few early years. Nowadays MAVEN is difficult to trick because of a pre-endgame engine that is sensitive to defensive considerations, and an omniscient endgame engine.

Can programs play correctly when holding a lead? It is a good question, but the answer is in the eye of the beholder. In the olden days players considered that there was only one valid approach to holding a lead: *shut down the board.* In part through MAVEN simulations and published analyses, players now realize that many techniques are available for holding a lead. For example, you can simply outscore your opponent (MAVEN's favorite). Alternatively, you can turn over tiles to end the game faster. Or you can shut down the board. Or you can try to "fill" the board—opening the board more quickly can lead to a faster shutdown in the end. Or you can create specific tactical situations too numerous to even conceive of.

Does MAVEN play correctly when holding a lead? It seems to play well. Nevertheless, we suspect that it would be better if its simulations returned winning percentage in addition to point spread.

How about coming from behind? Can MAVEN do this? It is hard to say. On the one hand, MAVEN has always been adept at winning no matter how desperate the situation. On the other hand, this may be because its opponents do not play well while leading. The previous paragraphs describe humanity's slavish devotion to mechanically blocking when holding a lead. In some games, MAVEN exploited this single-minded pursuit by climbing back into contention and winning even without a bingo.

To answer the question, MAVEN really has no specific technique for handling deficits. It seems to play pretty well, but that is because by maximizing point differential it is highly likely to maximize winning percentage as well. Again, one suspects that adding a winning percentage evaluation to the simulator would produce better play.

Going further, is it possible to build a computer program that plays perfectly as a practical matter? MAVEN is getting close to this standard. The implementation needs tweaking to obtain the property of "asymptotic practical perfection," and we need to add a winning percentage evaluator. Those changes alone might do it. It might happen faster with a few other techniques described in Chapter 11.

14.2 Championship Caliber Midgame Evaluation Function

MAVEN's Basic evaluation model combines a simple rack evaluator with a triple-word square evaluator. Because of the expansive nature of the developmental stage of a Scrabble game, this simple model is remarkably robust. It significantly improved the methods that human experts employed through the late 1980's. When combined with exhaustive move generation, the Basic evaluator yields championship-caliber play in the midgame.

14.3 Pre-endgame Analysis Using Probability-Weighted Search

MAVEN's probability-weighted search engine addresses the pre-endgame, which is the stage of the game when the possible opposing racks are completely enumerable in practice.

MAVEN's search engine allows the program to understand defensive considerations that it would otherwise be unable to understand. This can result in prescient plays that block the specific bingo that an opponent intends to play. On the downside, it can result in losing games that would have been won with normal offensive play.

The literature on games describes similar engines used in other games. The closest parallels involve bridge and backgammon. Bridge is like Scrabble in that it involves hidden information. It is unlike Scrabble because in bridge there is no stochastic behavior during the play. Probability-weighted search in bridge, therefore, has a different flavor [56]. In bridge, the program must make inferences about the location of cards, as the distribution of hands is vitally important. In Scrabble, it appears that inferences are not so important.

Therefore, the closest parallel to MAVEN's use of probability-weighted search is in backgammon. Backgammon program employ shallow selective searches of continuations for evaluating moves [38]. The difference between Scrabble and backgammon is that in backgammon the probability distribution of the continuations is known, whereas in Scrabble it must be assumed. This difference does not appear to be important in practice.

MAVEN's implementation could stand refinement, but is still an important part of MAVEN's complete system.

14.4 Endgame Analysis Using the B* Algorithm

The endgame in Scrabble begins when the bag is empty, at which point the state space has perfect information. MAVEN includes an endgame search engine and evaluation function to handle such situations.

The endgame engine appears to be unique for its speed and quality of play. In the decade since MAVEN's debut, the only contender the author is aware of is the exhaustive search engine of ACBOT [25].[54] MAVEN's level of performance in endgame tactics is the standard by which all players, human or computer, are judged.

The endgame engine uses the B* search algorithm. The endgame of Scrabble is one of the few domains for which B* appears to outperform all other techniques.

14.5 Lookahead Using Monte Carlo Simulations

MAVEN contains a Monte Carlo search engine for exploring the non-deterministic portions of the search space. MAVEN uses the term "simulation" to refer to this capability.

This engine leverages the static evaluation functions and improves upon them. Of particular importance is the ability of the simulator to rationalize the evaluation of moves that were generated by different static evaluation functions. This capability allows the engine to explore moves that focus on different aspects of the position, while maintaining an integrated view of the whole problem.

Evaluating moves using simulation produces the strongest level of Scrabble play known to date. Careful measurement of game data suggests that the error rate of MAVEN's current implementation is less than 6 points per game, with easily fixed bugs accounting for half of the total.

Simulation is not unique to MAVEN, as at least five other engines (QUETZAL, BOBBOT, ACBOT, PIOBOTJR, and Steven Gordon's unnamed program) employ the technique. The author believes that MAVEN was the first to deploy an effective system of move selection using real-time simulations. However, the literature is sketchy, so it is difficult to be certain. Nevertheless, this thesis does represent the most complete description of a sophisticated simulation system published to date.

14.6 Revolutionized Scrabble Positional Theory

MAVEN's development contributed to the development of Scrabble positional theory. Before the advent of MAVEN, it was impossible for human players to fully appreciate the subtleties of positional evaluation. The differences between the mean values of individual tiles, for example, can be less than half a point, whereas the standard deviation of the score of a single move is about 20 points. This large signal-to-noise ratio obscured the true values of evaluation parameters until the advent of computers.

[54] An exhaustive engine is inexorable when it can complete its analysis within time constraints, but that is not guaranteed.

MAVEN employed automated self-play to determine the optimal values of evaluation parameters. Later, MAVEN's simulation capability extended humanity's ability to test hypotheses about the game.

By distributing MAVEN itself in demo form, by publishing MAVEN's analyses of positions, and by inspiring the development of other simulation systems, we have significantly improved the understanding of Scrabble among tournament players. The result has been the wholesale revision of Scrabble positional theory. Concepts that were dogma among master players have been discarded in favor of techniques that MAVEN has proven. The process continues to this day, as human masters use simulations to explore the space.

Glossary

ACBOT—Strong Scrabble program developed by James Cherry. ACBOT has gradually added analytic capabilities over the years. Accessible in online game rooms.

Anamonic—A phrase that contains those letters (and only those letters) that make a bingo with a given stem.

Anchor—In move generation, an anchor square is the leftmost empty square covered by a move that is adjacent to an occupied square.

B*—Search algorithm developed by Hans Berliner. Applied in MAVEN's endgame engine.

Bag—Where the tiles are stored until they are drawn.

Basic Evaluation Model—A humanly-executable rack evaluator, devised by Nick Ballard, Charlie Carroll, and the author. In this thesis, we have used the Basic model for expository purposes, even though MAVEN's actual rack evaluator differs in particulars.

Bingo—A play that uses 7 tiles from the rack, scoring a 50-point bonus.

Bingo line—A spot on the board where a bingo may be played. Bingo lines require seven empty squares and some contact with the board. Normally requires a minimum of contact with the board.

Bingo out—A bingo played in the endgame (which necessarily finishes the game). This massive blow normally amounts to 100 points after the opponent's tile penalty is figured in. It occurs in about 5% to 10% of all games.

Blank—A tile that has no letter on its face. A blank may stand for any letter. It scores zero points, yet is the most valuable tile of all because of its flexibility.

Block (noun)—A move that mechanically prevents a threat.

Block (verb)—To move so as to prevent an opponent's threat.

Bonus—Synonymous with "bingo." "Bonus" is the preferred term outside of North America.

Cheat Sheet—A single page showing the most important words in the game, published by the NSA. In most clubs, newcomers are handed the Cheat Sheet on their first visit, and then allowed to play with the Cheat Sheet in view (hence the name) for a period of time.

CRAB—Early Appel and Jacobson program modified for Computer Olympiad play by Steven and Graeme Thomas. Still available in source form through the Web.

CROSSWISE—Scrabble program by Jim Homan. Winner of two Computer Olympiads. Apparently similar in philosophy to early versions of MAVEN.

DAWG—Directed Acyclic Word Graph. This is the data structure at the heart of Appel and Jacobson's move generator.

DLS—Double Letter Square.

Double-double. AKA four-timer—A move that covers two double word squares, thereby scoring four times the sum of the tile values.

Draw—The tiles taken out of the bag to replenish the rack.

Drek—Terrible tiles.

Dump—A move that rids the rack of poor tiles without regard to score. A low-scoring replacement for an exchange.

Duplicate—Two identical tiles.

DWS—Double Word Square.

Early game—The phase of a game in which the possible openings increase with every new word placed.

Endgame—The phase of the game after all tiles have been drawn from the bag. The endgame is a deterministic phase.

Extension—A move that adds more than one letter to a word on the board.

Fish (noun)—A play that hopes to draw specific tiles from the bag.

Fish (verb)— To make a fishing play.

GADDAG—A data structure derived from DAWG that is used by Steven Gordon's move generator.

Going Out—To use all of one's tiles in the endgame, ending the game.

Handler Function—In the move generator, each move is passed to a called-supplied handler function for processing. This architecture separates the responsibility for generating moves from the processing of moves after they are generated.

Heavy tiles—Tiles with a face value of 3 or more. In English, BCFHJKMPQVWXYZ.

Hook—A letter that can be added either to the front or back of a word to make a longer word. Also a move that exploits a hook.

Hotspot—A place on the board where high scores are possible.

Inference—A conclusion reached by a player after considering the implications of the opponent's recent moves on the opponent's rack.

JQXZ—A word that contains either J, Q, X, or Z. Most such words are short, and they occur frequently.

Middle game—Synonymous with Early game, but it would not include first turn situations.

NSA—National Scrabble Association. In North America the NSA sanctions tournaments and club play.

NSC—National Scrabble Championship. The largest tournament in North America. The NSC is held every two years, alternating with the Worlds. Now a 31-round event spanning 5 days with a significant prize fund.

One-tile—To play out a game one tile at a time when the opponent is stuck with awkward tiles that do not play.

Opening—A hotspot or bingo line.

OSPD—Official Scrabble Players Dictionary. This book is published by Merriam-Webster in North America for use by home players as a reference dictionary. Shortcomings of the OSPD with respect to tournament play prompted the creation of the TWL98.

OSW—Official Scrabble Wordlist. This list of words was the reference list in the United Kingdom until the year 2000, when they switched to SOWPODS.

Out—To use all of your tiles in an endgame.

Out Bonus—The premium collected when playing out, equal to twice the sum of the opponent's tiles.

Out-in-one, out-in-two, etc.—To play out in an endgame in one move, two moves, etc.

Overlap—A move that forms more than one crossword.

PIOBOTJR—A recent program by John Babina that distributes simulation workload over a network of computers.

Playing Out—To use all of one's tiles in the endgame, ending the game.

Pre-endgame—The phase of the game just before the endgame.

Prefix—The start of a word. Normally refers to a common letter group (usually 3 letters) that occurs at the start of a large number of words.

Premium—A scoring bonus.

Premium Square—A square on the board marked with a premium, e.g., double letter square, triple word square, etc.

QUETZAL—A program by Tony Guilfoyle and Richard Hooker. QUETZAL is said to have employed simulation for selecting moves, but this does not seem to have improved its tournament results.

Rack—The tiles that a player holds.

Rack Evaluator—A heuristic function for evaluating the quality of the tiles left on the rack after a move.

Rating—A measure of a player's skill, computed by the NSA based on tournament games.

Rollout—Synonymous with simulation. The term "rollout" is used in backgammon.

Separation—In an endgame search using B*, a move achieves separation when B* proves that the lower bound on that move's evaluation is higher than the upper bound of all other move's evaluation.

Setup—A move that arranges tiles to create a profitable follow-up play.

Simulation—A process that repeatedly plays out a situation for a number of turns, with the goal of exposing the situation's dynamic aspects.

SOWPODS—The dictionary consisting of the combination of the TWL98 and the OSW. Called SOWPODS because it was originally a combination of the OSPD and OSW, and Scrabble players cannot look at seven letters without trying to make a word out of them.

Stem—A set of 6 or 7 letters that make a bingo with a large number of tiles.

Stuck—Unable to play out in an endgame because a tile does not go down.

Suffix—The end of a word. Normally refers to a common letter group (usually 3 letters) that occurs at the end of a large number of words.

TD—Temporal differences, a learning algorithm developed by Richard Sutton.

Telegraph—A move or other behavior that allows the opponent to infer information about your tiles.

Tile—If you have gotten this far, then you must know what a tile is.

Tile Density—A rack evaluation factor that adjusts the values of tiles according to the number of each type that remains in the bag.

Tile Pattern—A set of tiles that have an associated value in the rack evaluator.

Tile Tracking—The process of noting which tiles have been played, so that a player can know which tiles remain.

TLS—Triple Letter Square.

Triple-triple. AKA nine-timer—A move that covers two triple word squares, thereby scoring nine times the sum of the tiles.

Triple Word Line—An opening to a TWS consisting of a single tile in line with the TWS and having no interference from the board.

TSP—Precursor to CROSSWISE. By Jim Homan. Twice Computer Olympiad Champion.

TWL98—The word list used by North American players. Published by Merriam-Webster, and only available to NSA members because it contains offensive words.[55]

TWS—Triple Word Square.

TYLER—A strong program by Alan Frank. TYLER performed well at three Olympiads, never finishing more than a few games behind the winner.

Unistem—A set of 6 or 7 letters for which only one letter makes a bingo.

Unseen—Tiles that are either in the bag or on the opponent's rack. So-called because they cannot be seen.

Volatility—An intuitive measure of how easy it is to come from behind.

Vowel dump—A move that plays more vowels than consonants. Used to restore the vowel / consonant balance of a rack.

Worlds—The largest international tournament, that crowns a World Champion. An invitational event in which invitations are apportioned based upon a country's Scrabble population. Also an existence proof that competing financial interests can work together when they share a common goal. The Worlds uses the SOWPODS lexicon. The Worlds are held every other year, alternating with the NSC.

WSC—The Worlds.

[55] I am shocked, shocked, to find offensive words in here!

Index of People

Alexander, Steven—Scrabble master. Maintainer of the Scrabble FAQ. Has long used TYLER to annotate every game he plays.

Avrin, Paul—New York area expert.

Babina, John—Scrabble programmer. Author of PIOBOTJR.

Ballard, Nick—Scrabble master. Backgammon World Champion, 1982. Also 5 Dan in Go, and USCF chess master, but you probably guessed that. Ballard published *Medleys*. Ballard used MAVEN extensively for simulation and practice.

Berg, Mark—New York City resident who has a knack for beating much stronger opponents. Is the weakest player ever to beat MAVEN in a tournament game.

Braud, Conrad—Surprise high finisher in 1989 NSC. His game Braud-Tiekert was published in SPN. See notes in Appendix B.

Cappelletto, Brian—National Champion 1999, runner-up in 2000. World Champion 2001. First Scrabble prodigy. Became a top player by age 15. (Prodigies are older in Scrabble because you have to be able to read, I suppose.)

Carroll, Charlie—Minnesota master renowned for efficient study habits. Invented anamonics. Carroll used MAVEN extensively for simulation and practice.

Cherry, James—Author of ACBOT. Expert player represented Canada at 2001 Worlds.

Chew, John—Scrabble expert. Member of the NSA Dictionary committee. John Chew graciously provided MAVEN with its officially sanctioned TWL98 word list. Chew implemented a postprocessor ("poslfit") for MAVEN's simulation log files.

Dixon, Jan—Scrabble expert is frequently the highest-rated woman. Has a knack for pre-endgames. Played in MAVEN's first tournament, and contributed a win to MAVEN in its third tournament.

Edley, Joe—1980, 1992, and 1998 National Champion, along with more tournament wins than any other player. SPN editor. Vice President of the NSA. Always among the highest rated players. Big booster of MAVEN.

Felt, Robert—One of the first masters to embrace MAVEN's rack evaluation theories. Longtime master. Felt dominated the 1990 NSC, winning by 3 games over nearest rival. Regularly makes the American team for the Worlds.

Fisher, Stephen—Montreal master renowned for creative setups and artful phonies.

Frank, Alan—Scrabble master and author of TYLER. Was a prominent Boston-area tournament director. Frank devised the original NSA rating system. Still active on the crossword game programmer's newsgroup.

Geary, Jim—Colorful master player. Renowned for fishing for and then playing the nine-letter WATERZOOI. Geary published a newsletter for a number of years.

Gibson, David—South Carolina master won NSC in 1994. Gibson also won the Superstars tournament, the richest tournament ever held.[56]

Gordon, Steven—Scrabble master and mathematics professor. Steven invented the GADDAG data structure. Early work on selecting moves using simulations.

Graham, Matt—Scrabble master. World runner-up in 1997. New York City area standup comic. MAVEN user.

Halper, Ed—New York City area Scrabble master.

Hersom, Randy—Scrabble master. Saw Edley's threatened ZIGGURAT, but then...

Homan, Jim—Author of CROSSWISE and TSP. Contributed to the NSA's rating system.

Kramer, Jim—Minnesota master. Kramer contributed a stunning number of anamonics. MAVEN user.

Kreiswirth, Rose—New York area expert. Dealt MAVEN a loss in its first tournament.

Logan, Adam—1996 NSC Champion. Second Scrabble prodigy. Mathematician. MAVEN user.

Lund, Richie—Flamboyant master from Brooklyn, NY. Regular top-5 finisher in NSC.

Mead, Jeremiah—Top master, who lives in the Boston area. Was a longtime Latin teacher in the school system of my hometown.

Morris, Peter—Top Canadian/American player. Hard to say which. Morris won 1989 NSC and 1990 Worlds. Very early MAVEN user.

Neuberger, Jim—New York area master. Frequently a top finisher in early NSC events, but never won. Neuberger contributed two wins to MAVEN's first tournament, but beat MAVEN with a superb pre-endgame fish in its third tournament.

Norr, Rita—New York area master. 1989 National Champion. Author of a book of Scrabble puzzles.

Odom, Lisa—Minnesota master. Odom represented the US at several Worlds.

Pellinen, Steve—Minnesota master. MAVEN user.

---

[56] It must be noted that "richest" is a relative concept. Prizes in Scrabble tournaments are not large enough to justify the effort spent to maintain skill at the level necessary. The players are not in this for the money.

Polatnick, Steven—American Scrabble master from Florida. Steven represented the USA at WSC'2001, finishing fourth.

Pratt, Dan—Frequent top finisher in early NSC events.

Root, Steven—A Massachusetts expert who first evaluated MAVEN.

Sherman, Joel—Winner of 1997 Worlds. New York area master. Sherman graduated from Bronx HS of Science in the same year as the author. Heavy MAVEN user.

Southwell, Charlie—A master from Washington, DC.

Tellis, Forrestt—Colorful, friendly expert. Has a penchant for fishing.

Tiekert, Ron—1985 NSC winner. New York area master, transplanted to Atlanta. One of the game's greatest players. Invented simulation.

Tierney, John—New York Times reporter. Tierney arranged a match pitting MAVEN against Joel Sherman and Matt Graham.

Toal, Graham—Computer Scrabble developer. Graham operates a web site (www.gtoal.com) that has comprehensive listings of computer Scrabble resources.

Wapnick, Joel—1983 NSC Champ, and 1992 runner-up. 1999 World Champ and 1993 and 2001 runner-up. Wapnick is a noted author. Music professor. Renowned for word knowledge. Wapnick won NSC Brilliancy prizes in 1990 and 1992, so his positional skills are sharp, too. Early MAVEN user.

Watkins, Mark—BOBBOT author and Scrabble expert.

Watson, Robert—1988 NSC Champion.

Wolfberg, Mike—A Massachusetts expert who first evaluated MAVEN. Dealt MAVEN its first loss in a tournament game.

References

[1] Sheppard, B., (1997) Neural Nets in Backgammon, in *Backgammon Magazine*, B. Robertie (ed.).

[2] Kopec, D. (1990) Advances in Man-Machine Play. *Computers, Chess, and Cognition* (ed. T.A. Marsland and J. Schaeffer). Springer-Verlag, New York. ISBN 0-387-97415-6.

[3] Neumann, J. von and Morgenstern, O. (1944). *Theory of Games and Economic Behavior*, Second Edition 1947. Princeton University Press, Princeton.

[4] ChessBase (2002). Junior 7 defeats Mikhail Gurevich (FIDE 2641) by 3.5-0.5. http://www.chessbase.com/events/2002corinth4.htm

[5] Carroll, C. and Ballard, N. (1991). Efficient Study Habits, in *Medleys*, Issue #5, May 1991.

[6] Ravichandran, C., (2000). LeXpert, a software program designed to assist word game enthusiasts in expanding their knowledge of words. Available online at http://www.carolravi.com/LeXpert/Default.asp.

[7] Ballard, N., (ed.) (1992). The Citizen Kane Problem, in *Medleys*, Issue #23, November 1992.

[8] Wapnick, J. (1986). *The Champion's Strategy for Winning at Scrabble® Brand Crossword Game*. Stein and Day, New York. ISBN 0-8128-6247-3.

[9] Weissman, A, and Weissman, D., *Letters for Expert Game Players*. Issue unknown.

[10] Howell, D. C. (1989). *Fundamental Statistics for the Behavioral Sciences*. PWS-Kent, Boston. ISBN 0-534-91694-5.

[11] Aho, A.V, Hopcroft, J.E., Ullman, J.D., (1974). *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.

[12] Levy, D.N.L. and Beal, D.F., (eds) (1990). Scrabble Tournament Report. *Heuristic Programming in Artificial Intelligence, The Second Computer Olympiad*. Ellis Horwood, New York.

[13] Turcan, P.J. (1982). A competitive Scrabble program, *SIGART Newsletter 80* (1982). Reprinted in M.A. Bramer (Ed.), *Computer Game-Playing: Theory and Practice*, Ellis Horwood Ltd, Chichester, UK, pp 209-220.

[14] Shapiro, S.C., (1982). Scrabble crossword game-playing programs, *SIGART Newsletter 80* (1982). Reprinted in M.A. Bramer (Ed.), *Computer Game-Playing: Theory and Practice*, Ellis Horwood Ltd, Chichester, UK, pp 221-228.

[15] *Official Scrabble Players Dictionary, Third Edition*, Merriam-Webster, 1995.

[16] Sheppard, B. (1990). *MAVEN's Endgame Ability-a Comparative Analysis*. Sheppard Company, Concord, MA.

[17] N. Ballard, B. Sheppard, and C. Carroll, 1983. *Renaissance of Scrabble Theory 2*. In *Medleys* Issue #17, August 1992.

[18] Edley, J. and Williams, J.D. (1994). *Everything Scrabble*. Simon and Schuster, New York. ISBN 0-671-86686-9.

[19] H.J. Berliner, The B* tree search algorithm: A best-first proof procedure. *Artificial Intelligence*, 1979, Vol. 12, No. 1., pp. 23-40.

[20] Baczynskyj, B. and Welsh, D.E. (1985), *Computer Chess II*. Wm. C. Brown, Dubuque, Iowa. ISBN 0-697-09911-3.

[21] Andrew W. Appel, Guy J. Jacobson, "The World's Fastest Scrabble Program", *CACM*, Vol. 31, No. 5, May 1988, pages 572-578,585.

[22] Steven A. Gordon, A Faster Scrabble Move Generation Algorithm, *Software Practice and Experience*, 24:2, Feb 1994, pp.219-232.

[23] Berliner, H.J. and Ebeling, C. (1989). Pattern, Knowledge, and Search: The SUPREM Architecture. *Artificial Intelligence*, vol. 38, no. 2, pp. 161-198.

[24] Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P., (1983). Optimization by Simulated Annealing, *Science*, 220, 4598, pp 671-680.

[25] James Cherry's web site, http://www.doe.carleton.ca/~jac/scrab.html

[26] Wapnick, J., (1998). *A Champion's Strategies*. Self-published CD-ROM.

[27] Ballard, N. (ed.) (1992). Consensus Game #2, Move 10, in Medleys, Issue #19, July 1992.

[28] Steven A. Gordon, A Comparison Between Probabilistic Search and Weighted Heuristics in a Game with Incomplete Information AAAI Fall 1993 Symposium on Games: Playing and Learning, *AAAI Press Technical Report FS9302*, Menlo Park, CA

[29] Buro, M. (1998). From simple features to sophisticated evaluation functions. In H.J. van den Herik and H. Iida, editors, *Proceedings of the First International Conference on Computers and Games (CG-98)*, volume 1558 of *Lecture Notes in Computer Science*, page 126, Tsukuba, Japan, 1998. Springer-Verlag, Heidelberg.

[30] Sutton, R., (1988). Learning to predict by the method of temporal differences, Machine Learning 3, pp 9-44.

[31] Watkins, M. www.math.uga/~mwatkins/fgame01.pdf. Analysis of a game between David Boys and Sam Kantimathi using BobBOT simulations.

[32] Ozag, J. and Lawrence, M. (1987). *The Ultimate Guide to Winning Scrabble Brand Crossword Game*. Bantam Books, New York. ISBN 0-553343068.

[33] Yedwab, L. (1985). *On playing well in a sum of games*, Technical Report MIT/LCS/TR-348, MIT, Cambridge.

[34] Berliner, H.J., (1979). On the Construction of Evaluation Functions for Large Domains, *6th International Joint Conference on Artificial Intelligence*, pp 53-55.

[35] M. Chrochemore and R. Vérin, 1997. *On Compact Directed Acyclic Word Graphs* in *Structures in Logic and Computer Science*, a selection of essays in honor of A. Ehrenfeucht, J. Mycielski, G. Rozenberg and A. Salomaa, eds., LNCS 1261, Springer-Verlag, 1997. pp 192--211. Available online: http://www-igm.univ-mlv.fr/~mac/REC/97-CSA.html.

[36] Ballard, N. (ed.) (1992). Consensus Extra #17, Ballard-van Leunen, in *Medleys*, Issue #16, April 1992.

[37] http://members.ozemail.com.au/~aspa/koala/strategy.htm

[38] Tesauro, G. (1995). Temporal difference learning and TD-GAMMON, *CACM* 38 (3) pp. 58-68.

[39] Chew, J. personal communication (1996).

[40] Edley, J. (ed) (1990) For Intermediates: Z spots. *Scrabble Players News*, no. 82. Williams & Company, Greenport, NY.

[41] Slate, D.J. and Atkin, L.R. (1977). Chess 4.5—The Northwestern University Chess Program, in *Chess Skill in Man and Machine*, P. Frey (ed.), Springer-Verlag, New York, pp 82-118.

[42] McAllester, D.A. (1985). *A New Procedure for Growing Mini-Max Trees*. Technical Report, Artificial Intelligence Laboratory, MIT, Cambridge.

[43] Palay, A.J. (1983). *Searching With Probabilities*. Report CMU-CS-83-145. Carnegie-Mellon University, Pittsburgh, PA.

[44] Berliner, H.J. and McConnell, C. (1995). *B* Probability Based Search*. Carnegie-Mellon University Computer Science research report, Pittsburgh, PA.

[45] Sheppard, B., (1990). *MAVEN's Endgame Ability—A Comparative Analysis*. Sheppard Company, Concord, MA.

[46] *Webster's Collegiate Dictionary, Tenth Edition*, Merriam-Webster.

[47] Uljee, I. (1992). Letters beyond numbers. In H.J. van den Herik and L.V. Allis *Heuristic Programming in Artificial Intelligence 3, The Third Computer Olympiad*, pp. 63-66. Ellis Horwood Limited, Chichester, England.

[48] Jim Homan's web site. http://www.cygcyb.com/catalog/software.html#CrossWise

[49] Gordon, S., (2002). Global analysis, posted to newsgroup wordgame-programmers. Available online at http://groups.yahoo.com/gourp/wordgame-programmers/message/332.

[50] Lerman, J., (1993). The Rating System—A Closer Look, in *Medleys*, issue #36, December 1993.

[51] National Scrabble Association web site, available online at http://www.scrabble-assoc.com/tourneys/2001/build/notes/25.html

[52] Wapnick, J. (2002). Personal communication.

[53] Sheppard, B., (2002). World-championship-caliber Scrabble. *Artificial Intelligence* 134, pp 241-275.

[54] Frank, I., Basin, D., and Matsubara, H., (1998). *Monte-Carlo Sampling in Games with Imperfect Information: Empirical Investigation and Analysis*. Research report of the Complex Games Lab, Ibaraki, Japan.

[55] Galperin, G. and Viola, P. (1998). *Machine Learning for Prediction and Control*. Research Report of the Learning & Vision Group, Artificial Intelligence Laboratory, MIT, Cambridge.

[56] Ginsberg, M. (1999). GIB: Steps Toward an Expert-Level Bridge-Playing Program, in *Proc. IJCAI-99*, Stockholm, Sweden, pp 584-589.

[57] Billings, D., Pena, L., Schaeffer, J., and Szafron D. (1999). Using Probabilistic Knowledge and Simulation to Play Poker. In *AAAI National Conference*, pp 697-703.

[58] Kopec, D., and Bratko, I. (1982). The Bratko-Kopec Experiment: A Comparison of Human and Computer Performance in Chess. *Advances in Computer Chess 3*, M. R. B. Clarke (ed.), Pergamon, Oxford, pp 57-72.

[59] Gordon, S.A., (2002). Opening Library. Posting to newsgroup wordgame-programmers. Available online at http://groups.yahoo.com/group/wordgame-programmers/message/423.

[60] Cherry, J., (1997). Posting to newsgroup crossword-games-pro@mit.edu. Preserved at http://members.aol.com/icenine378/jaconsim.txt.

[61] Tiekert, R. (1989). personal communication.

[62] Ballard, N., personal communication.

[63] Ballard, N. *Anatomy of an Endgame*. Self-published monograph.

[64] John Chew's website, http://www.math.toronto.edu/~jjchew

[65] Frank, A. (2002). Inferences. Posting to newsgroup wordgame-programmers. Available online at http://groups.yahoo.com/group/wordgame-programmers/message/317.

[66] Edley, J. (ed) Championship Play. *Scrabble Players News*. Williams & Company, Greenport, New York.

[67] Elo, A.E. (1966) *The USCF Rating System*. U.S. Chess Federation

[68] Sheppard, B., (1993) Opacity, in *Rack Your Brain* #5.

[69] Babina, J., (2002). PioBotJr Current Stats. Posting to newsgroup wordgame-programmers. Accessible online at http://groups.yahoo.com/group/wordgame-programmers/message/310.

[70] Sherman, J. (1998). Personal communication.

[71] Graham Toal's web site http://www.gtoal.com lists many speculative proposals for improving the performance of simulations.

[72] Chew, J. (2002). *The Canonical List of Anamonics*. Available online at http://www.math.toronto.edu/~jjchew/scrabble/anamonics.html.

[73] Carroll, C. (1993). Personal communication.

Appendix A – Rules of the Game

MAVEN plays the game under tournament rules, which differ slightly from the rules printed in the game box but not in any way that alters the fundamental nature of the game.

Tournament games have two players who alternate turns. Play begins by placing all 100 tiles into a bag. In the English language, the tiles have the distribution and point values shown in Table A-0-1.

Tile	Points	Number
Blank	0	2
A	1	9
B	3	2
C	3	2
D	2	4
E	1	12
F	4	2
G	2	3
H	4	2
I	1	9
J	8	1
K	5	1
L	1	4
M	3	2

Tile	Points	Number
N	1	6
O	1	8
P	3	2
Q	10	1
R	1	6
S	1	4
T	1	6
U	1	4
V	4	2
W	4	2
X	8	1
Y	4	2
Z	10	1

Table A-0-1 Tile Values and Distribution

Each players draws a single tile from the bag to determine who will move first. The player drawing closer to the start of the alphabet moves first, with a blank superseding all other tiles. The players return their single tiles to the bag, and shuffle. Then each player draws seven tiles and places them on the rack.

Moves consist of placing tiles on the board to form words. On the first turn, the word is constrained to pass through the central square. On turns after the first, the word is constrained to cover a square adjacent to a tile already on the board.

To make a turn, a player transfers tiles from his rack to the board to create a word. The tiles must lie within one row or column of the board, with no empty spaces intervening between the tiles played. It is allowable for preexisting tiles to intervene.

All words formed by a move are intended to come from a previously agreed reference dictionary (see Table A-0-2). A player may challenge his opponent's previous move if he believes that the opponent used a phony word. A neutral third party adjudicates the challenge. If the challenge is upheld (i.e., the move used a phony word) then the opponent's move is removed from the board and the tiles played are returned to the opponent's rack. The opponent scores zero for his turn. If the challenge is denied (i.e., all words were acceptable) then the challenger scores zero for his turn.

Geographic Region	Standard Reference Dictionary
North America	Tournament Word List ("TWL98")
Rest of the World	Combined North American and UK dictionaries ("SOWPODS")

Table A-0-2 Relation of Lexicon to Geographic Region

If at least seven tiles are in the bag, then a player may use his turn to exchange tiles. The player does this by discarding from 1 to 7 tiles, face down. He then replenishes his rack with new tiles from the bag. He then returns the discarded tiles to the bag. Exchanging tiles scores zero points for a turn.

A player may pass his turn at any time. He simply announces "Pass" and accepts a score of 0 points for his turn. He neither draws nor discards tiles.

A player's turn is complete when he announces his score. He then draws one tile from the bag for every tile he played, up to the number of tiles in the bag.

The score of a move is the sum of the scores of each word created by the play. The score of a word is basically the sum of the face values of the tiles used to create it, but premiums apply.

If a tile covers a double or triple letter premium square then the value of that tile is doubled or tripled.

If a tile covers a double or triple word premium square then the value of the whole word is doubled or tripled. The premiums from covering double or triple letter squares apply before doubling or tripling the word's score. Note that the central square is a double word square. If a turn covers two word premium squares then the score is doubled or tripled twice.

Note that premium squares apply only to the moves that first cover them. If that tile is later used by a move then that tile counts at face value.

A blank tile may be used as any letter, but it is scored as zero points. When a player places a blank he must announce what letter the blank stands for. That blank stands for that letter for the rest of the game.

If a move uses all seven tiles then it scores an additional bonus of 50 points.

The game ends when a player uses all of his tiles and there are none left in the bag, or when each player scores zero points for three consecutive turns. If a player uses all of his tiles then he scores a bonus of double the sum of the face values of the tiles remaining on the opponent's rack. Otherwise, the game ends with both players still having tiles left. Then each player is penalized by subtracting the sum of the face values of the tiles left on his own rack.

The player with the higher score is the winner. Ties are possible.

Appendix B – Annotated Games

The *Scrabble Players News* (SPN) originally published the first two games. MAVEN analyzed the games using simulations, and the author wrote notes summarizing the data. Joe Edley edited these and published the result.

The notes presented here differ from the SPN notes in several respects. The author wishes to illuminate how computer play differs from human play, without minding that the notes occasionally make humans seem completely clueless. Such a point of view would not have gone over well in the pages of SPN. Consequently, these notes emphasize disagreements between MAVEN and Edley, whereas the SPN notes smoothed over such differences.

These notes also present occasional situations where MAVEN is clueless. As is often true of computer programs, in these situations MAVEN extensively optimizes the wrong quality. Usually, this means that MAVEN is maximizing point differential at the expense of winning percentage.

The third game is from the AAAI-1998 match between Adam Logan and MAVEN.

The fourth game is from the recent World Championship final match. That game uses the SOWPODS lexicon. Scoring is higher in SOWPODS games, especially because there are many big plays using heavy tiles.

I owe many thanks to Joe Edley for giving me the opportunity to annotate in *Scrabble Players News*. Thanks also to Joel Wapnick, who proofed these notes.

B.1 Braud-Tiekert, NSC 1989

This game was the first that the author annotated for *Scrabble Players News*. Editor Joe Edley let me write all of the notes myself, and even introduced me as "one of the foremost strategists of the game."[57]

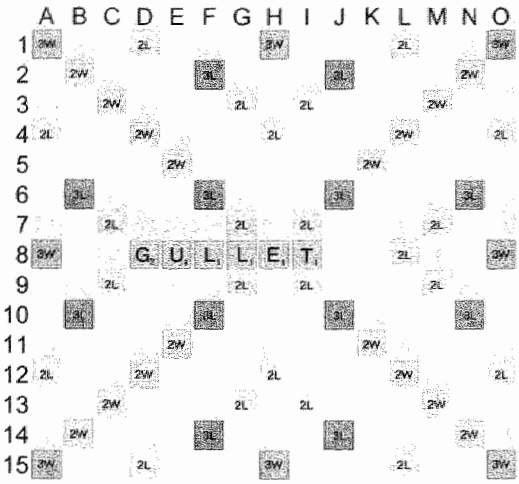
The game was played in the 1989 National Championship. Ron Tiekert was the 1985 National Champion, and the player who had achieved the highest rating of all time (2167). His opponent was Conrad Braud, who at a rating of only 1600 had basically no business competing at this level. However, every championship features a player who performs vastly beyond his expectation, and in 1989, it was Braud's turn. As a result, Braud had the privilege of playing Tiekert, and having a game published in SPN.

Braud's initial rack was EGLLNTU. His GULLET (8D, 18, N) was clearly better on general principles than GLUTEN (8D, 18, L) because of the superiority of N over L. However, Joel Wapnick reported that simulations using his version of MAVEN prefer GLUTEN. This is not surprising because simulations are not guaranteed to distinguish very close plays. My simulations prefer GULLET.

[57] I am really just a programmer, but if Joe says so...

Tiekert's "GYROID" (D8, 22, EL) is phony. He may have thought "GYROID" was acceptable, or else he figured that Braud would not challenge. That was a good call, since Braud did not challenge. Unfortunately, Tiekert missed ELYTROID (I5, 67).

Braud's rack was ?AEINVY. Braud found the lovely NAIVETY (9H, 68). The other bingos were VENIALLY (F3 or F2, 73) and NATIVELY (F2, 73). Best seems to be VENIALLY (F3, 73) because it reduces bingo comebacks by overlapping more of "GYROID". It is characteristic that humans will find seven-letter bingos but miss eights.



Tiekert: D, E, I, L, O, R, Y, 0

Braud's last: GULLET (8D, 18) 18

After Braud's GULLET (8D, 18)

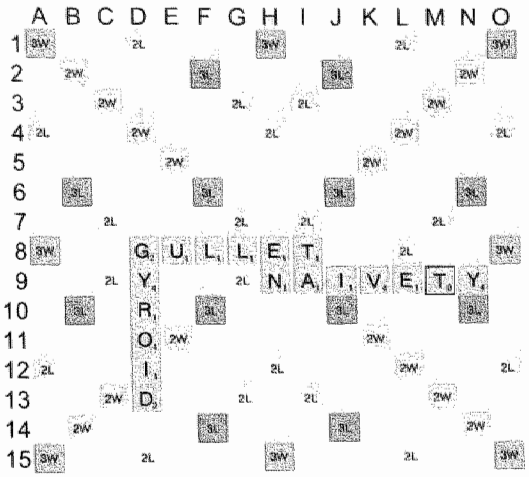
Braud suspected that GYROID was phony, but was not sure so he decided to play his bingo. Weaker humans characteristically do not challenge when they hold good tiles, and will challenge when their tiles are poor. From a logical perspective this is, well, illogical. Such insecurity is less apparent in stronger players.

Tiekert played FEEBLE (L8, 30,HX), which is a strong move. One would not expect EX (E10, 38, BEFHL) to be as good as FEEBLE, since leaves like BEFHL are muscle-bound. However, these tiles can land excellent scores in this situation. For example, if Tiekert drew an A then he would have FLAB (D11, 39) and HEAL (8L, 48). Simulation shows that EX is 1.5 points better than FEEBLE.

This game was played under OSPD1. In TWL98, the play would be HEBE (8L, 55, FLX), as pointed out by Joel Wapnick. But don't rush off to the latest edition of the OSPD to look up HEBE, because you will not find it. HEBE is an ethnic slur, and the OSPD excludes such words. This is one reason why the tournament players have adopted TWL98.

Braud missed the big play KAROO (14J, 50, AS) because he did not know the word. His play was KOLA (F6, 18, AORS). This suggests that Braud missed the R hook, too, since KOR (14J, 29, AAOS) is also better than KOLA. Humans often miss hooks, except for back –S hooks. Charlie Carroll told us that he deliberately creates front hooks, figuring that the spot is positionally neutral, but his opponent is likely to miss it [73].

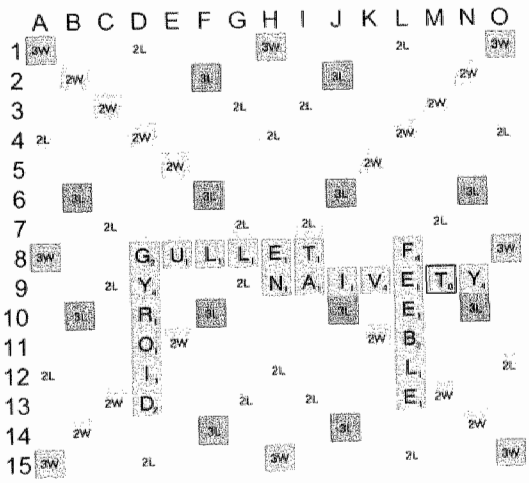
Tiekert, holding CEHIRUX, benefited from Braud's error with XERIC (14J, 72, HU). Please note the defensive value of offensive play. Braud could have prevented Tiekert's big move by taking that spot himself. This is a frequent occurrence.



Tiekert: B, E, E, F, H, L, X, 22

Braud's last: NAIVETY (9H. 68) 86

After Braud's NAIVETY (9H. 68)

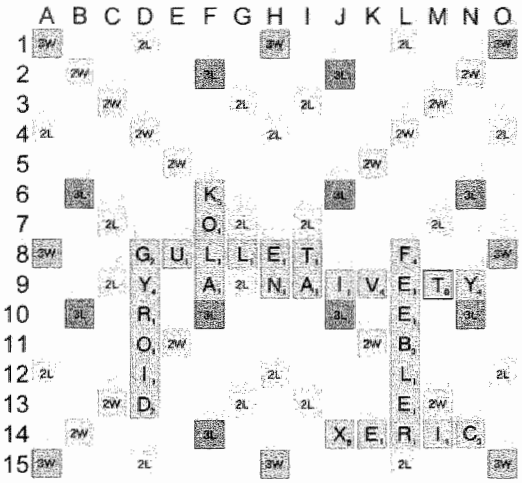


Braud: A, A, K, O, O, R, S, 86

Tiekert's last: FEEBLE (L8, 30) 52

After Tiekert's FEEBLE (L8, 30)

Braud's HO (13I, 18, AJORS) attempted to set up a hook at 13H for his O or R. (Either that, or Braud was unaware of the risk he was taking.) This type of setup is only successful in restricted situations. Besides holding a hook tile, there must be a low probability (less than 50%) that the opponent holds one, or the board should have another hotspot to draw the opponent's attention. These conditions are not met here. HO takes M, O, R, T, and W as front hooks, so there are 17 unseen tiles that hook.



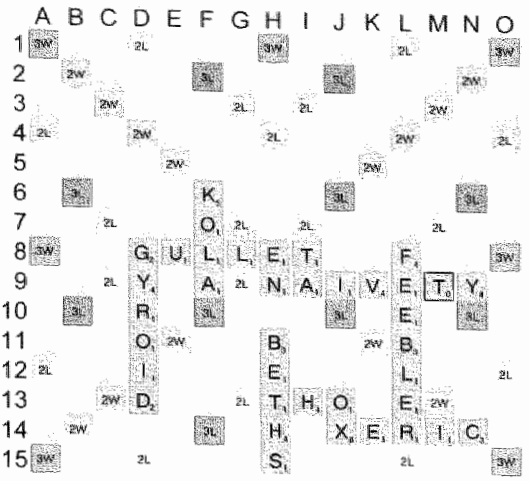
Braud: A, H, J, O, O, R, S, 104
Tiekert's last: XERIC (14J, 72) 52

After Tiekert's XERIC (14J, 72)

There is no need for a setup. Clearly stronger is JO (13I, 26). Also stronger is HAJ (C12, 37), despite leaving a hook for HAJI on row 15. Braud told Edley that he prefers JO, which is typical. Allowing the opponent to hook HAJ would be too much for most humans to bear. Yet, Braud did essentially the same thing inadvertently!

Tiekert, holding BEEHSTU, pounced on the opening with BETHS (H11, 39, EU). My original notes to the game in SPN simply state that BETHS is best. My simulations back this up, but the difference is small. BETH (C9, 31, ESU) is just 2 points inferior. I believe that BETH would actually be better were it not for the fact that BETHS takes the TWS. Wapnick's simulations actually placed BETH above BETHS.

Braud played JAB (11J, 24, AOPRS), but he had better: JAR (10B, 26, AOPRS). JAR hangs a hook in the A-column, which is just what I castigated Braud for last turn, but here it is correct because the fundamental requirements are met. Braud keeps an A for AJAR, so he has the key tile. Moreover, Tiekert has less than a 50% chance of holding an A. We know that because he did not keep an A last turn (otherwise BATHE instead of BETHS), so Tiekert had only 5 draws at 5 out of 55 tiles. This is obviously less than a 50% chance.



Braud: A, A, J, O, P, R, S 122

Tiekert's last: BETHS (H11, 39) 163

After Tiekert's BETHS (H11, 39)

JAR is about 7 points better than JAB, and works especially well for Braud, the player who trails and is rated 400 points below his opponent. Braud should seize the opportunity to land a score that could turn the game around.

Tiekert held EINOOQU. Tiekert's QUEY (N6, 36, INOO) is one of the best moves. The alternative was QUOD (13A, 28, EINO). Comparing offensive values, QUOD is better, because it undoubles the O's and keeps an E instead. This is obviously worth 8 points. However, the defensive aspects favor QUEY. Tiekert should not open a new bingo line (the B-column) and instead should close the N-column. This is correct tactically because his rack leave is not likely to benefit from either B-column or N-column openings (which have letters—U and Y—that go badly with vowel-heavy racks). Additionally, blocking is strategically correct because Tiekert is leading. These plays are close, but on balance, QUEY is the winner.

Braud held AOPPRST. Braud's PROPS (15D, 12, AST) was a big error. Braud should try to score some points. POT (C12, 20, APRS) is my choice. POT keeps tiles for RASP (15A, 33) if nothing better offers. A setup for a 33-point move is not valuable, especially since the setup burns the S. The real benefit is that the S hook means that Braud gets a good score next turn if he draws heavy tiles, and if he does not draw heavy tiles then Braud's bingo chance would be excellent.

Braud's move displays his lack of confidence in GYROID, since POPS (14A, 28, ART) is obviously better than his play, except that it hooks GYROID. Good call, except that the best time for distrust in GYROID was on turn 2!

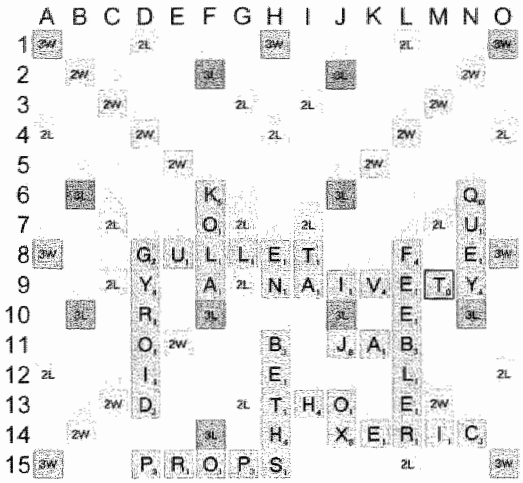
POP (C12) and PAP (C12) score more than POT (C12), but they must be rejected because they leave nasty hooks: POPE and PAPA. PORT (C12, 22, APS) could be best.

Joel Wapnick writes [52], "I'm not crazy about plays that block the row 15 bingo line. But POT is probably right. If I were further behind, perhaps PORT (10B)." PORT would create a hook for SPORT in the A-column.

Tiekert played TIROS (10B, 20, ?NO), missing the bingo STOOKING (6B, 63). According to Edley, Tiekert probably saw STOOKING but was not certain it was legal.

Braud held AEILNST, and did not miss his chance: LANKIEST (6C, 64), the only playable bingo.

Tiekert answered by also finding the only playable bingo: NATATION (B8, 64). He had a choice of where to place his blank. It can be either N. He chose to place the blank on B8, where it is adjacent to the triple-word square that Braud is most likely to take. Since Braud took the spot immediately, Tiekert's decision saved 3 points.



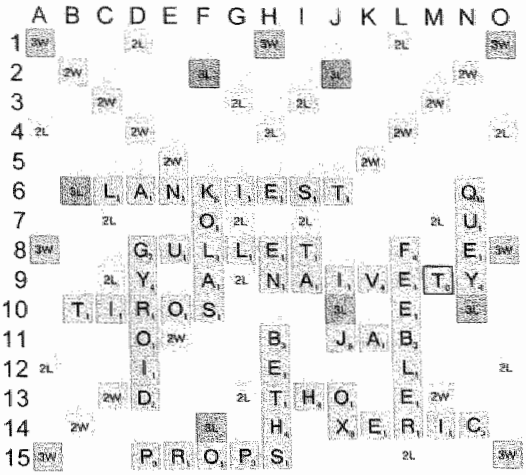
Tiekert: ? I N O O S T 199

Braud's last: PROPS (15D, 12) 158

After Braud's PROPS (15D, 12)

Incidentally, MAVEN is not guaranteed to place the blank optimally, since it will only generate one of the two placements of the blank. MAVEN's algorithm guarantees only that if there is a difference in score then MAVEN will place the blank so as to maximize score. Here both placements of the blank score the same, so MAVEN will arbitrarily choose one.

In theory, MAVEN should generate moves that play blanks even when the corresponding tile is on the rack. In practice, the principle of saving the blank for the future is ironclad, so MAVEN is correct to only generate moves that *require* the blank. This situation is the only case the author is aware of where this practice is potentially incorrect.



Tiekert: _ A A I N O T 219

Braud's last: LANKIEST (6B, 64) 222

After Braud's LANKIEST (6B, 64)

Braud played FROM (A6, 34, ARV). He either missed or did not know FORAM (A5, 37, RV).

Tiekert's rack was ADMNNTU, with the score 283-256 in his favor. This is one of my favorite positions. Tiekert's move was UNMADE (H1, 30, NT). MAVEN's Basic evaluator considers UNMADE to be much better than other moves. Nevertheless, two other moves are worth considering: MAUND (5H, 24, NT) and DAUNT (5H, 19, MN).

The great strength of DAUNT is that it mechanically blocks the board, and it would be extremely hard to open it back up. DAUNT allows only 25 bingos to play using the unseen tiles AACDDEEEGGIIINORRSUVWWZ. By contrast, UNMADE allows 343 bingos to play through the M alone!

Braud: A F M O R R V 222

Tiekert's last: NATATION (B8, 64) 283

After Tiekert's NATATION (B8, 64)

DAUNT is also equal to UNMADE with respect to preventing a big Z play from H1, since there are no six-letter words ending in DE that use the Z and the unseen tiles.

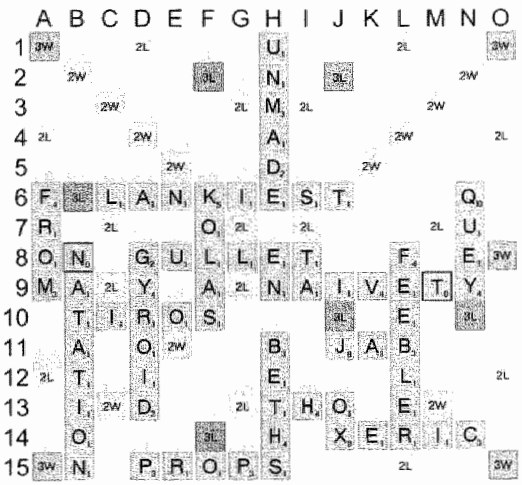
But the decision is not obvious because DAUNT scores 11 points less than UNMADE. Tiekert may need those points because DAUNT only takes a 46-point lead.

When I published the original notes, I favored DAUNT, but I started having second thoughts before the ink was dry on the paper. Publishing this opinion must have caused a stir, since I heard about it from Tiekert and from Richie Lund, both months after publication. Tiekert restricted himself to the comment that the decision was not at all clear. Richie's comments started with, "Who are you to criticize a move by Ron Tiekert..." and I had better not print the rest.

Besides, it really is not clear. Tiekert may need 11 points. Maven's simulations show that DAUNT is probably not best, as it finishes behind MAUND and UNMADE regardless of the assumptions made by the simulation. But I have gotten either MAUND or UNMADE as the simulation winner by changing parameters. For instance, MAUND finished on top when simulating to the end of the game using inferences and winning percentage.

Braud would do well to play AZO (F13, 32, DERUV) instead of AZURE (4H, 28, ADV). AZURE helps Tiekert to block the top lines. AZO keeps them open. ZANDER (2F, 38, UV) is highest scoring, but that rack leave is dreadful, and ZANDER also helps the opponent close the board.

The SOWPODS play ADZE (A12, 68, RUV) reminds us that defensive considerations must be made relative to the concrete possibilities of the position. In OSPD1, Tiekert's UNMADE may be an error because it gives Braud a chance to play a bingo. Yet in SOWPODS, UNMADE could be best because Braud would have the chance of a big Z play.



After Tiekert's UNMADE (H1, 30)

Tiekert held CEGGINT, and his move GENIC (2F, 18, GNT) iced the game. Besides being Tiekert's biggest move, GENIC shut down the remaining bingo lines, eliminating Braud's comeback chances.

Braud's OW (14E, 28, ADIRV) is one of the best. My only suggestion is to play AW instead. Normally one keeps A instead of O, but this position is exceptional because the bag contains an A but does not contain an O, plus the O can be used under the G in GENIC.

Holding ADEGNNT, Tiekert played FEED (8L, 25). My original notes in SPN rated ETNA (A12, 28) as best, but technical improvements to pre-endgame simulations in the intervening years overturn that assessment. FEED is best.

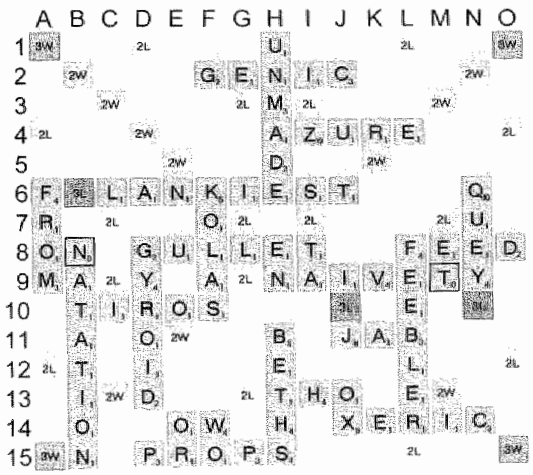
The endgame went as in Table 0-3, leading to a final score of 386-360 in Tiekert's favor. Note the characteristic inaccuracies of human endgame play. Braud's first turn dropped 11 points. Tiekert returned the favor by dropping 5. Braud gave back 1 point. This is typical; experts average 10 points of errors per endgame.

Word	Spot	Score	Comment
VIS	M2	+27	DIVA (D3) ETNA (A12) WARS (M1) = 25
GENIAL	C1	-14	ANGINA (D1) VAW (2M) ET(5K) = 14
WAG	1A	+21	WAIR (4A) = 19
TEN (DR) Total	2B	-10 -6 = 18	

Table 0-3 Braud-Tiekert Endgame

Braud: A, D, I, O, R, V, W, 284
Tiekert's last: GENIC (2F, 18) 331
11 Unseen tiles: ADEEGINNSTW

After Tiekert's GENIC (2F, 18)



Braud: A, D, I, R, S, V, W, 312
Tiekert's tiles: A, E, G, I, N, N, T, 356

After Tiekert's FEED (8L, 25)

Both players dropped well over 100 points on errors in this game. This is more than Tiekert usually drops, but was a good game for Braud. I usually figure 4 points of differential for every 10 rating points, so Braud's play would be typical of a player at about the 2150 – 100 * 10 / 4 = 1900 level.

An interesting point, which surprises many observers, is that the winner often makes more errors than the loser. This state of affairs is normal, in my experience. The player who draws better tiles often falls far short of his maximum potential, yet wins. The player with worse tiles has greatly constrained choices, and therefore makes plays that are closer to optimal. Prior to the advent of computer training partners, this aspect of Scrabble would cause humans to overestimate their skill, because when they examined their losses they discovered that there was little they could have done to improve their play. Nowadays, any player that genuinely wants to know where he stands can find out.

B.2 Adam Logan – Ed Halper, NSC 1990

Before Adam Logan was National Champion, he was a 15-year old wunderkind who took the Scrabble world by storm, achieving a top-30 ranking before his 16th birthday. Moreover, he was a MAVEN user. A coincidence, perhaps? You be the judge. Logan's opponent in this game is Ed Halper, an expert of long standing who finished third in the 1990 Championship.

Logan started the game holding DEGNPRT. Logan played TREND (8H, 16, PG). The conventional theory of the day was that you should play long words, other things being equal, since such plays help to draw the blanks. Nowadays we look askance at such plays, because they give the opponent many places to play 8-letter bingos. Long words are categorically bad, but length is a drawback that can tip the scales if longer and shorter plays are otherwise equal.

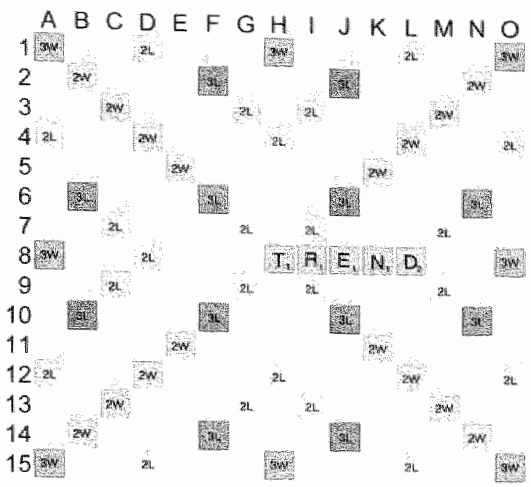
MAVEN includes a length penalty, but I find that it is ineffective. The penalty was computed by determining the amount by which MAVEN overestimated the value of the first turn as a function of length. While the measurement is correct, the feature is ineffective because it rarely changes a move decision, even though shorter words are often better than long words. I believe the reason is that many factors favor short plays, but MAVEN is projecting them onto a single dimension (i.e., length). For instance, it pays to keep bingoish tiles. It pays to give few through-letters to the opponent. It pays to deny overlaps, and deny access to word multipliers. These factors are projected onto a single dimension (length), whose correlation to the true evaluation result is low.

Another expert tendency is to overemphasize such possibilities as extending TREND using an ING to hit the triple word square at 8O. Such factors are minor. In many games, neither side will ever have an ING on the rack. Moreover, ING is just as likely to be on your rack as on the opponent's rack. Finally, TRENDING would only score 30 points, so it just is not worth preventing. One should actually be happy if an opponent used an ING to play TRENDING. Racks containing ING score 7 points more than average racks, whereas TRENDING scores 5 points less than average.

The real objection to TREND is that Logan has a better play. DREG (8F, 12, NPT) keeps better tiles, avoids opening a double word square and scores reasonably well. It simulates 3 points better than TREND.

Halper's JOIN (K5, 22, ISUX) is a good move, but there are better plays. The author likes JEUX (J7, 34, IIOS). IIOS is not the best rack leave, but you score 12 extra points and also dump the U. That JEUX is better is clear, even to a non-simulator.

An interesting play is JEU (J7, 10, IIOSX), and on this issue, Edley and I disagreed. Simulation results show that JEU is the best move in this position by 10 points, because of the chance of playing XI (10J, 59) or even bigger plays hooking the X onto JEU. Edley downgraded JEU and JEUX to equality. Edley's experience is that such plays do not work out as well as they should. For instance, Edley notes that JEU might tip off Logan that something funny is going on, and he might block. However, there are few ways to block.



After Logan's TREND (8H, 16)

This disagreement shows that simulations do not always resolve differences of opinion. At issue is the fact that low-scoring plays like JEU telegraph that something funny is going on. In the case of JEU, a reasonable inference is that Halper is dumping two awkward tiles to fish for a bingo. You would not think X because playing JEUX now scores 24 points more. Why would you figure Halper for holding an X? JEU + X-hook makes 69 points, whereas JEUX + Normal move = 34 + 39 = 73. Why would Halper tie himself down to this setup?

There are three reasons why JEU works for Halper. First is that it gives Halper a way to split up his II pair without losing any points. Because of the II, JEUX + Normal Move is likely to be 34 + 25 rather than 34 + 39. Second, it pays to delay placing the X on the board for a turn because having the X on J10 would create profitable dumps for Logan on the K-column. Finally, it pays to block the only open vowel on the board, since if Logan is clogged with consonants then he will have no place to play.

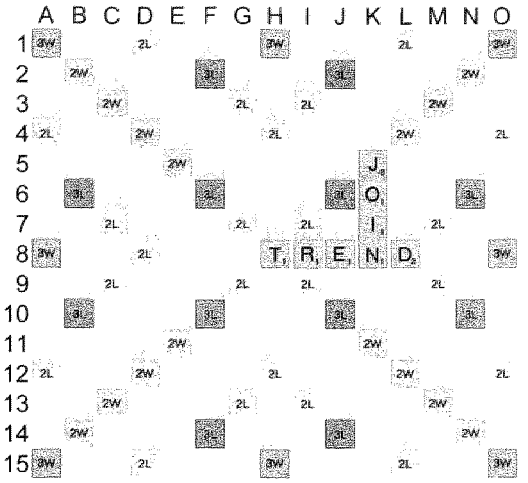
Logan played HAET (J6, 30, EGNP). Logan missed PATHOGEN (6G, 74) and HEPTAGON (6E, 70). PATHOGEN is best because of the higher score.

Halper held AIPSUXY. Halper's biggest spot is the Y back hook of TREND. Halper has a choice between PIXY (M5, 36, ASU) and PYX (M7, 38, AISU). PIXY scores two points less, but plays an I. Normally, clearing an I is insufficient compensation for losing two points, but here it is because the rack already has a preponderance of vowels, and the IU combination plays poorly.

We are duly impressed that Halper was not swayed by the hotspot that PIXY creates on N6. Computerized positional studies show that such spots are insignificant, but the humans of the day thought otherwise.

Logan exchanged all 7 tiles, but should definitely not have. He has the two dumps HOPING (6I, 15, EGNR) and PHONING (6H, 16, EGR), and these scores and racks are better than exchanging. Logan must have missed the spot, which is not surprising because it is hard to see moves through three separated tiles.

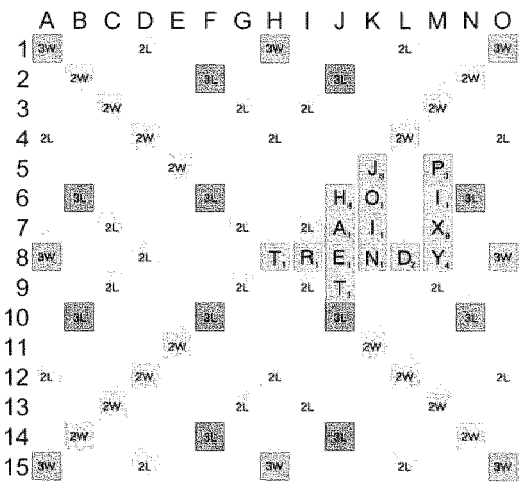
By the way, HOPING is about 4 points better than PHONING, which the author never would have guessed. The reason is that the EGNR rack leave has good bingo prospects through the T in TREND, and if you play PHONING then you block that spot. For example, if you draw an A or I from the bag then the H column plus your tiles would be AEGNRT or EGINRT, and you just need two reasonable tiles to complete the bingo.



Logan: A, E, G, H, N, P, T 16
Halper's last: JOIN (K5, 22) 22

After Halper's JOIN (K5, 22)

Passing 7 tiles was a poor choice of tiles. It is 5 points better to keep ER, as suggested by the Basic rack evaluation parameters. However, in those days players only kept blanks and S. I remember one game in which MAVEN exchanged, keeping one tile, which sent the opponent into defensive spasms, blocking all sorts of possibilities. Late in the game the opponent drew a tile from the bag, stared at it with astonishment and said, "I thought MAVEN held the last S. What did MAVEN keep when it exchanged?" He realized that his defensive posture had been completely wrongheaded, and perhaps cost him the game. When he found out that MAVEN kept an E, he was disdainful. "Why keep an E? Chances are you will draw one from the bag!" However, that player is wrong on several counts. First, odds are that you will *not* draw an E from the bag, and then your bingo chance is hurt badly because 61% of all bingos contain an E. Second, if



Logan: E, G, G, N, N, P, R 46
Halper's last: PIXY (M7, 36) 56

After Halper's PIXY (M7, 36)

you draw a second E you are still better off with EE than with no E's at all. Finally, it was folly to sacrifice many points to defend against an S-hook that would cost less to allow. I understand why he was cross, though I have to laugh about it.

Halper held AEESSTU. Halper correctly played one of his duplicate S's, but DUETS (L8, 12, AES) is weaker than USE (10I, 15, AEST) by 3 points in score and also in rack leave. The move that simulates best is JUPE (5K, 13, AESST), which beats USE by 4 points. The point is that JUPE creates a hook on the O-column.

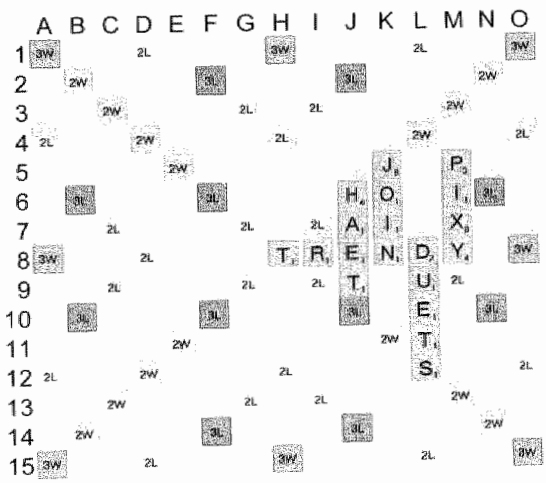
Here Edley and I disagree again. Edley doubts that JUPE has winning chances as good as USE. Edley feels that if the opponent holds an S or blank (34.6% of the time) then the opponent will get a big score on the O column. I figure that the opponent will *not* have the key tiles 65.4% of the time, and then Halper is sure to make a big play because he *does* hold the key tile. When the opponent has the big tiles there might not be a big play; the opponent could have a blank and a bingo elsewhere, or be unable to use the hook and still hit a triple word square.

Also relevant is that if Logan has a blank and no bingo then he really does not hurt Halper. For example, in the actual game Logan drew ?BDEHTV from the bag, so he could play BETHS hooking JUPE for 52 points. I say it is no big deal. Sure, 52 points are a lot, but *he played a blank and got little for it*.

This disagreement contrasts MAVEN's icy willingness to put the game on the line with a big play versus Edley's instinct to stretch the game out. Edley's strategy may be appropriate for playing against weaker players. Against weaker players, it is poor strategy to allow a crusher. MAVEN is tuned for ideal theoretical play (i.e., for beating masters), which often makes experts cringe.

Logan played BETH (11J, 18, ?DTV), which has the benefit of blocking the 10J square where an S may be used as a hook. Logan considered that DUETS telegraphed that Halper retained an S, since why else would he play an S for only 12 points? So he thought blocking was in order.

Edley insisted that BETH (11J), as played by Logan, was the best play, despite MAVEN simulations that backed BETH (10K, 17, ?DEV). It is obvious why MAVEN likes BETH (10K); keep an E, dump a T, and better vowel / consonant balance. Plus, BETH (10K) puts an S-hook in the O-column, where Logan may be able to use his blank. But Joe insisted that MAVEN was underestimating the inference that Halper retained an S. Because of this, I actually



After Halper's DUETS (L8, 12)

implemented an inference engine while preparing this thesis. Unfortunately, the inference engine is not able to confidently predict that Halper kept an S when he played DUETS, so I have nothing more to add. Joel Wapnick believes that Edley is wrong, and so do I, but there is no way to prove it with the information that I have.

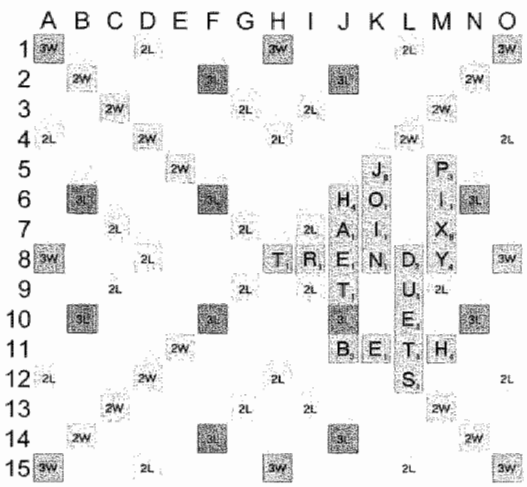
The natural inferences from Halper's DUETS are that Halper retained an S and a T. There are three moves involved in these calculations.

1) USE (10I, 15) is obviously better than DUETS, with the only difference being that DUETS plays a T. Therefore, the retention of another T is indicated.

2) DUET (L8, 5) is 7 points less than DUETS, whereas it keeps S. Since an S would normally not be played for only 7 points of compensation, this indicates that an S was kept.

3) JUPE (6K, 13) is obviously better than DUETS but for an S and T.

The point is that drawing inferences is risky business, since Halper did keep an S and did not keep a T. If MAVEN concluded that Halper kept a T, then it might retain that notion until Halper played either a T or a bingo, or until MAVEN drew the last T out of the bag, or Halper makes a move that allowed MAVEN to infer that Halper did not have a T. In the case of inferring a T, the downside is probably unimportant, but what if MAVEN inferred that the opponent kept a Q?



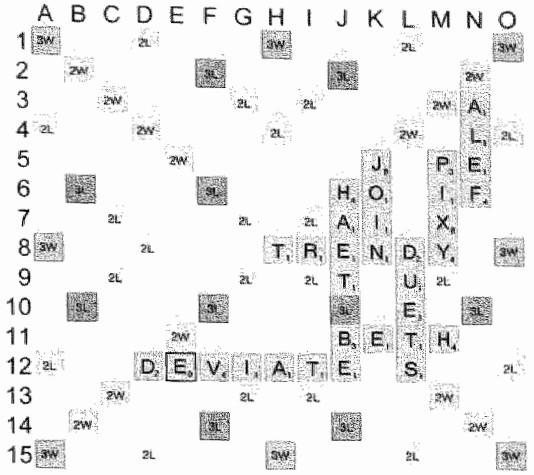
Halper: A, A, E, E, F, L, S 68
Logan's last: BETH (11J, 18) 64

After Logan's BETH (11J, 18)

Halper played ALEF (N3, 32, AES). ALEF is best, of course. A surprising simulation result: LEAF (N3, 32, AES), which plays in the same position for the same score, also has the same simulation result. Now you might not be surprised at this, but a Scrabble expert would be because LEAF leaves the E adjacent to O4, so there is the likelihood of a 40-point reply from O1 to O4, putting a consonant on O4 where it will score eight-fold. Simulations show that ALEF's vowel-consonant-vowel-consonant pattern is easier to overlap than LEAF, so the O-column is often used for *longer* plays after ALEF than after LEAF. The result is about equal.

Logan played DEVIATE (12D, 76). VISTAED (N9, 83) is clearly better, but Logan missed it. VISTAED is a tough find, despite ending in –ED. Logan considered VITAE (O1, 37, ?D), but only momentarily. The bingos are much better.

I would have spent all afternoon looking for a bingo in Halper's rack, AEINRSW. But there is none. Halper played REWAN (O1, 40, IS). REWIN is better, as keeping A instead of I is usually better. And REWINS (O1, 49, A) is better still, since an S is worth 7.5 points, and REWINS scores 9 points more. Edley restrained me from saying that REWINS is best, saying that keeping the S is significant because whoever bingos next will be in the driver's seat. To which I retort that as played Halper keeps the S and Logan is still in the driver's seat. The OSPD, Second Edition, added the word RAWIN (a type of radar), which is the best play nowadays.



Halper: A, E, I, N, R, S, W, 100
Logan's last: DEVIATE (12D, 76) 140

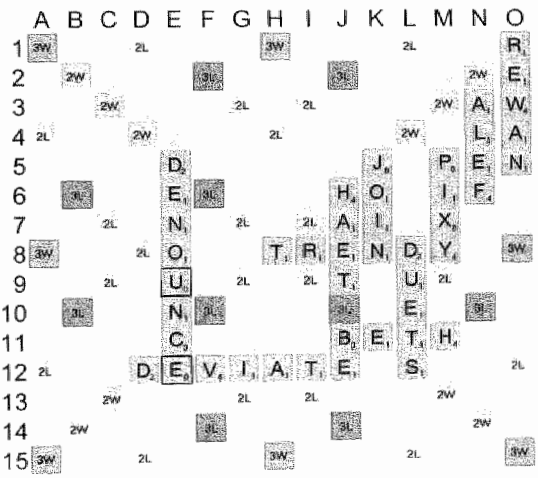
After Logan's DEVIATE (12D, 76)

The last three moves are a good example of how opening triple word squares is not as dangerous as it seems. Logan had first crack, but was unable to exploit the opening because he had a better play elsewhere. It is a good thing for Halper that the opening was there, since it allowed him to keep pace.

Logan found the lovely double-double DENOUNCE (E5, 86) out of the rack ?CDENNO. A trifle better is CONDENSE (E5, 86) in the same spot, probably because of restricting overlaps on F6.

Halper played the plausible phony "VIBRO" (8A, 33, FNS) against his young opponent, and Logan let him get away with it. Logan should not challenge unless he is confident that VIBRO is phony, since Logan's lead is large.

This play is a good example of an artful phony. The word is not a bingo, and it does not take the lead from behind, so the incentive to challenge it is smaller. However, it scores better than any real word, it takes a spot where Logan could score well, and it cleans out the rack.



Halper: B, F, I, N, R, S, V, 140
Logan's last: DENOUNCE (E5, 80) 226

After Logan's DENOUNCE (E5, 80)

Logan's non-challenge is interesting. I daresay that if I had tried to wing VIBRO past him then he would have challenged with confidence. The point is that there is a huge difference between an unfamiliar word played by a master like Halper, and one played by a patsy like me. This type of reasoning is very important to humans in judging the plausibility of unfamiliar words.

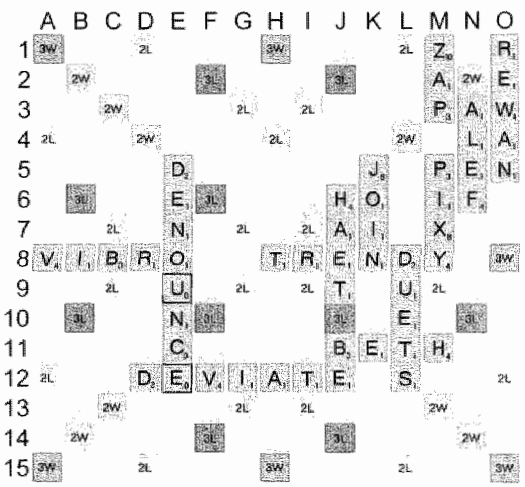
The best legitimate play is BARFS (H11, 30, INV). Halper is probably about 15 points better off having gotten away with VIBRO. Who says you cannot play better than MAVEN?

Logan's ZAP (M1, 44, GMSY) is best. Or perhaps GYP (M1, 34, AMSZ) is barely better. The justification for GYP is that the Z will score well on this board. For example, you need just an A or E for ZAMIA or MAZES at H11. However, Logan and Edley feel that leaving a G on row 1 is courting trouble in the form of a late –ING bingo.

This time Edley is definitely correct. The bird in the hand is worth a lot here. When you are leading, you should take points to sew up the game. GYP may get back the 10 sacrificed points on average, but when it does not then you place the game at risk. Moreover, the ING bingo consideration is real.

Halper's KEF (F4, 34, INOS) is good, and KIEF (F3, 35, NOS) is slightly better. But a surprising 5 points better still is OF (F5, 29, EIKNS). Keeping EIKNS proves to have big potential.

Logan held GLMOQSY. AGLY (H12, 24, MOQS) has better simulation results than Logan's move AMYL (H12, 27, GOQS) by 1.5 points. The difference is playing a G instead of an M, but you also lose 3 point in the trade. You would not expect an M to be worth 4.5 point more than a G, and I really do not know why it is so here. Possibly because there is a chance of a 48-point MOSQUE hooking BETH.



After Logan's ZAP (M1, 44)

The play that simulates best is HOMY (M11, 26, GLQS). It looks funny to keep all those consonants with the Q, but the point is that if Logan drew a U and an E or A then QUEYS or QUAYS (14J, 74) loomed. Here is a good example of how maximizing point spread does not always maximize winning chances. Logan, with the lead, should be concerned about playing off his Q with the greatest possible *speed*, not the greatest possible *score*. I am confident that AGLY wins most often.

Halper had LICENSOR (15H, 86), but his CORNIEST (H1, 86) is best because it leaves open the most bingo lines. CORNIEST opens bingos through the C and O along the top two rows, and leaves open bingos through the L in AMYL, and the –S hook on BETH. LICENSOR would have negate all of those chances, leaving only the T in TREND plus the letters of VIBRO. If Halper were leading after his bingo then LICENSOR should be better, but here CORNIEST is best.

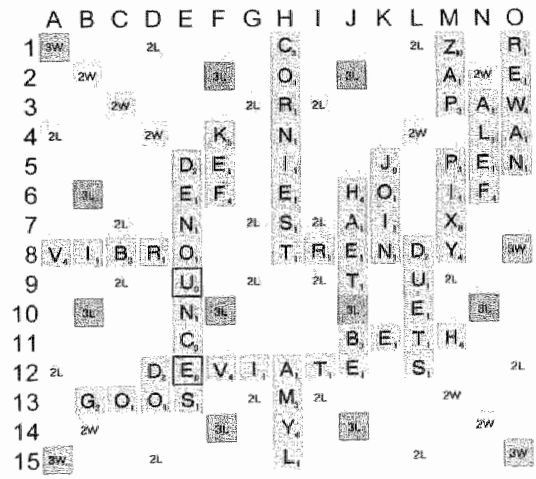
Halper: C, E, I, N, O, R, S, 207
Logan's last: AMYL (H12, 27) 297

After Logan's AMYL (H12, 27)

Joel Wapnick points out [96] that one advantage of CORNIEST is that it interferes with potential –IZER plays along the top row. I am not sure that such interference is in Halper's favor, but I am duly impressed by Wapnick's ability to spot such tactical subtleties.

Logan drew a U, and held GIOOQSU. Because of his U, he did not need to worry about playing his Q. Therefore, Logan's GOOS (13B, 23, IQU) was better than the biggest Q move, which was QUOIN (10A, 16, GOS). QUOIN kept the S, but at the cost of keeping the O and G, and a loss of 7 points.

Joel Wapnick liked GOO (13B, 11, IQSU) best. He commented [97], "Pick one of the A's for QUAGS (B10, 70) next turn." This is a nice insight, but simulations show that GOOS is better for both point differential (by 3 points) and winning percentage (by 5%). The move with the highest point differential is HOG (M11, 16, IOQSU). You just need an A for QUASI (14J, 75), and HOG scores more than GOO. But HOG is highly volatile, and the steady GOOS wins 3% more often, despite a point differential 1.4 points lower.



Halper: I, I, L, L, N, R, S, 290
Logan's last: GOOS (13B, 23) 320

After Logan's GOOS (13B, 23)

Many players would prefer QUOIN out of paranoia. Two particular neuroses afflict Scrabble players in this situation. "Quetoxipsychosis" is the irrational fear that it will be impossible to play a Q. In extreme cases, this causes a player to accept a 16-point move for a QU combination, simply to relieve the pressure. "Bingophobia" is the fear that the

opponent will bingo unless a player physically blocks. Acting on this fear causes frequent losses to opponents who never actually play a bingo. A genuine sufferer of quetoxipsychosis considers it to be a victory simply to play the Q away. Likewise, a bingophobic player "wins" if his opponent does not bingo. Perhaps the most horrifying sight in Scrabble is to witness two bingophobes playing against each other. Cover the children's eyes!

After Logan's GOOS, Halper needed to balance his rack, and MILL (13H, 15, INRS) does the best job.

Logan, holding AAEEIQU, missed EQUINE (10A, 37, AA). He played QUAIL (15D, 24, AEE). I speculate that this error was caused by focusing on positional goals rather than trying to find the biggest score. For example, Logan may desire to split up his AA combination, and may wish to block the 15th row. If he wished to find a play using the TLS at 10B, then he would have found EQUINE.

Since Halper's SURGING did not play, he needed a strategy for manufacturing a comeback. Halper's LUG (K13, 4, GINRS) is one of the best choices.

Halper's highest score is RIGGINGS (B7, 30, U), which leaves Halper only 9 points down. Now you might think that Halper would win many games since he is only 9 points down, but it is not true. Simulations show that Halper wins only 3% of all games after RIGGINGS. There are three drawbacks to RIGGINGS. First, it keeps a U, which is a poor tile. Second, it leaves four tiles in the bag, which means that the opponent will almost always play out first in the endgame (i.e., Logan plays 3 tiles, Halper empties the bag, and then Logan plays out in two unless Halper has a highly unlikely, unblockable out-in-one). Third, drawing six tiles to a U is unlikely to make a bingo.

Halper: G, G, I, N, R, S, U, 305
Logan's last: QUAIL (15D, 24) 344

After Logan's QUAIL (15D, 24)

LUG is more like it. LUG, VUG (A8, 7), BUG (C8, 7), GNU (10C, 6), and GUDE (5C, 6) all win by drawing bingo tiles to GINRS. Of these LUG is clearly best, since all of these moves bingo just as frequently, and LUG scores the most points.

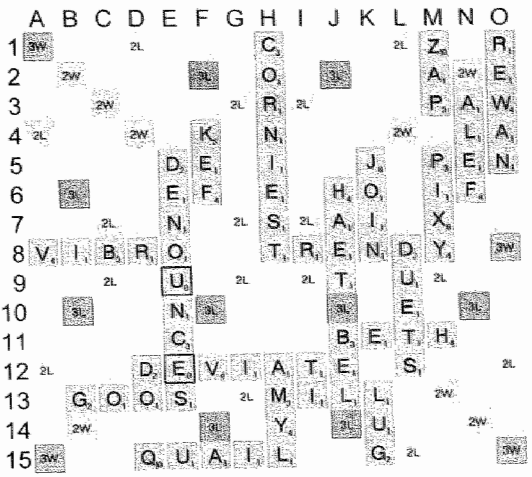
My notes in the SPN gave RUNG (10C, 9, GINS) as best, though it wins just 1% more often than GNU. After GNU, the opponent must block the most probable bingo lines, which are through the V and I in VIBRO. Then Halper has a 22% chance to bingo along row 1 with ORGANICS or GENERICS; along row 2 with ERINGOES, MISGROWN, NEGROIDS, or GENITORS; or hooking BETH with RUSTING, RESTING, or

GASTRIN. However, it turns out that Halper would not win after every bingo, and that cuts his winning chance.

After RUNG there is a new threat to Logan: Halper might hold the W for a big play from B10 to B14. For example, if Halper draws a W then he holds that tiles for WINGS (B10, 51). Logan must block the B-column, but he scores few points while doing so. Halper's bingo chances are the same, but he also wins 1% of the time when he draws the W and Logan's tiles are so bad that he cannot block.

But there is a problem with the whole analysis, which Joel Wapnick pointed out to me, and simulations quickly confirmed: Edley, MAVEN, and I had missed the best play entirely! Best is RUG (I8, 7, GINRS). RUG falls among the top 20 plays in MAVEN's list, but not the top 10. This accounts for MAVEN's failure to propose RUG. Edley and I have no excuse for missing it. Wapnick wrote, "RUG is interior, and blocks nothing, even partially." Quite right.

Simulations also dispute my analysis of RUNG. MAVEN's current simulators are more effective at pre-endgame decisions than the one it used when annotating this game originally, so I am not surprised that there is a difference. MAVEN says that RUNG drops 3% compared to RUG (I8), and LUG drops 1% compared to RUG (I8).



After Halper's LUG (K13, 4)

Unfortunately, this game was not destined to be a strategic masterpiece for Halper. Logan sewed up the game with the bingo ERADIATE (B4, 63). Most games are like this. The winner is often not player having the deepest strategic insight. The winner *is* often the player with the best move generation. The player that draws better tiles wins even more often!

After Logan's ERADIATE (B4, 63)

Halper fished bingo tiles (SWINGER), but they are a "nongo" because the word does not fit the board. He played GREW (A1, 45, INS), which is best. Logan played MOOT (C1, 18, IOU) out of IMOOOTU. Halper played out with GAINS (15K, 21+6). Logan won by 425 to 381.

This is a typical game in many respects. The players missed one bingo out of 5 opportunities, which is probably a little worse than top humans usually perform. The game also contained a phony and a missed opportunity to challenge.

In addition to the phony, both players made several errors of evaluation or move generation. These errors were small, but the accumulation was large. You can see how a computer would have an advantage over any human that is below the highest caliber. Perfect word knowledge and perfect move generation are invaluable in Scrabble. Players outside the highest level simply make too many errors.

This game shows how simulation makes a difference. Even a human expert who is thoroughly familiar with MAVEN's tile estimates would estimate that KIEF and OF from Halper's move 14 were about equally good. He would not think that OF had a 5-point edge. Simulations make that apparent.

At the same time, you can be amazed at the cognitive capacity of human beings. The players would have used about 20 minutes each in this game, and they made tremendous discoveries of plays involving words that normal humans have never even heard of.

B.3 Logan-MAVEN, AAAI-98, Game 9

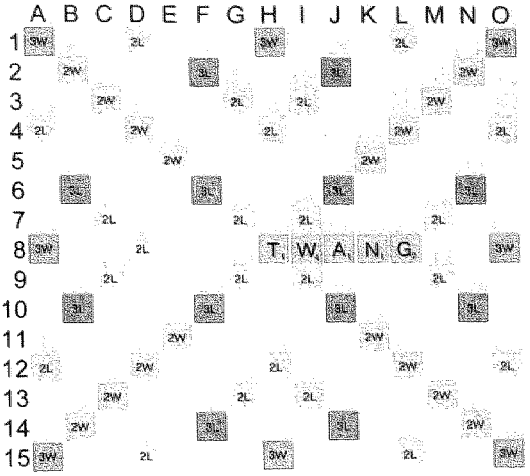
In a departure from tradition, I will present one of MAVEN's losses. In part, this is because I have published a win elsewhere [53] and I do not want to repeat myself.

Logan drew AGLNSTW, and started with TWANG (8H, 22, LS), which was his best play. Alternatives either do not play 5 tiles (E.g., GNAW) or burn the S for insufficient compensation (e.g., TWANGS (8C, 26, L)).

MAVEN's MORNING (L2, 20, T) is a normal move, and best.

Logan just needed an E last turn to have TWANGLES, which he undoubtedly saw. He drew an E, and held ELNNSSU. Now that he has drawn the E, he took the opportunity to slap down TWANGLES by extending TWANG.

TWANGLES is the highest-scoring move by far, but the low-scoring NUN (M1, 14, ELSS) is nearly as good. The ELSS rack leave has excellent bingo potential, whereas TWANGLES's NSSU rack is unlikely to make a bingo. NUN simulates within 0.1 points of TWANGLES.



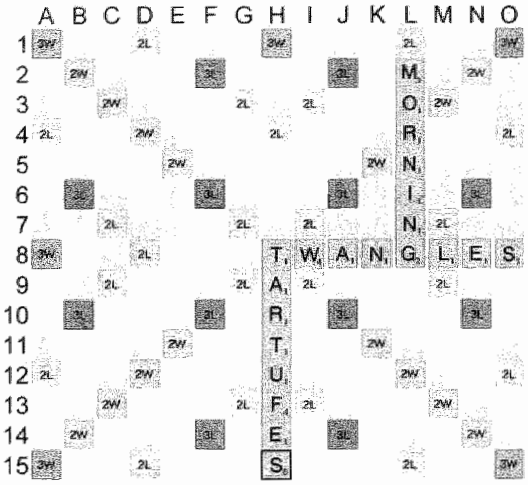
MAVEN:	I M N N O R T	0
Logan's last:	TWANG (8H, 22)	22

After Logan's TWANG (8H, 22)

MAVEN held ?AEFRTU. There were many bingos to consider from this great rack. Seven bingos score 83 points, all from either O1 (e.g., FEATURES) or H8 (e.g., TARTUFES). Any of these plays is acceptable, and MAVEN selected TARTUFES (8H). The best play is FRACTURES (O1, 83). FRACTURES simulates about 1 point higher than the other plays. It has a higher winning percentage because it allows fewer through letters, as pointed out by Joel Wapnick [52]. The 1998 MAVEN used a pruning trick to cut off simulations of bingos, and this feature pruned FRACTURES. The 2002 edition of MAVEN chooses correctly, thanks to its scalable simulation controller.

Here, Logan made a slight error. The best move according to the Basic evaluation model is ONYX (3L, 28, DNSU). Logan played NIXY (6K, 22, DNSU), which simulation calls a 2.4-point error.

Logan is concerned about opening the O-column with an X, as most humans would be. Simulation suggests that exposing the X is costly, but not as costly as giving up 6 points.



Logan: D, N, N, S, U, X, Y, 58
MAVEN's last: TARTUFES (H8, 83) 103

After MAVEN's TARTUFES (H8, 83)

MAVEN held ABEGHOO, a rack offering many profitable overlap plays. It is often hard to find the biggest move from such a rack, especially when the best play is not an overlap at all. The normal move of the Basic model is GOOMBAH (2I, 34, E). It is a strong move, but it has an unfortunate aspect: opening the O1 triple-word square (TWS). The O1 TWS can be used two ways: either parallel with short words like FA, HA, HE, etc., or through with words having H as second letter. The fact that there is a choice gives the opponent flexibility in using it. The spot is productive, since a simple two-letter dump like YE (1N) scores 35 points. GOOMBAH also opens the 1H TWS, but that is different because using the spot requires an A.

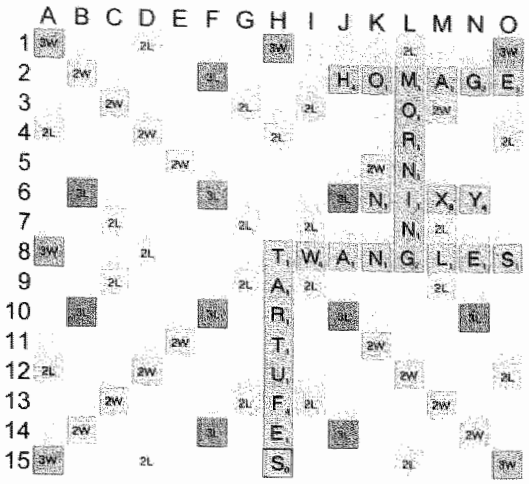
The fascinating move that MAVEN selects, which appears to be 2.5 points better than GOOMBAH is HOMAGE (2J, 40, BO). An obvious advantage of HOMAGE is that it scores 6 points more. Six points compensates for a weaker rack: BO instead of E is a disadvantage worth somewhat more than 6 points!

HOMAGE works better than GOOMBAH because it opens *two* triple-word squares. This sets up a tit-for-tat trade: the opponent takes one TWS, and MAVEN takes the other. This works out better than GOOMBAH, in which the opponent is likely to get one TWS, and the tiles are unlikely to cooperate so that MAVEN can take the other. Also, the two openings are about equal in value. Each is basically a direct (as opposed to overlap) opening that almost any rack will be able to exploit.

This shows how complicated Scrabble can be. Leaving one big opening is worse than leaving two big openings! At least in this case…

Recent simulations using winning percentage suggest that GOOMBAH is slightly better than HOMAGE, despite having a lower point differential. But this result is tentative, considering that the winning percentage evaluator is experimental technology at present.

Best is clearly SUNDERS (1D, 79), taking a TWS while scoring maximally. Many people have the misconception that it is better to leave both TWS openings, since it is set up as a trade. However, that is completely wrongheaded! Opening two spots may be better than opening one spot, but taking one is better than leaving two. Obviously, you want to take both spots if possible, and the only way to do that is to take one of them first!



Logan:	.D.N.R.S.S.U.	80
MAVEN's last:	HOMAGE (2J, 40)	143

After MAVEN's HOMAGE (2J, 40)

Logan's move, SUNDRESS (O8, 77), lost a couple of points of equity compared with SUNDERS. I doubt that Logan would evaluate SUNDRESS as better than SUNDERS. Yet, I also doubt that Logan missed SUNDERS. It is hard to guess what went wrong here. Maybe he misscored SUNDERS. It is hard to add scores correctly over the board. Many games contain an off-by-two error, from adding scores incorrectly. Another frequent error is an off-by-ten, attributable to an error in carrying while adding the scores of the moves. Some players routinely recheck the scores of any game decided by 10 points or less. However, Logan has a Ph.D. in Mathematics from Harvard, so arithmetic errors are unlikely. Is it possible that Logan forgot that SH is a valid hook for H? We will never know...

The reader can see how access to triple word squares works out in high-level games. MAVEN opened the spots, but Logan had to play his bingo. Then MAVEN got a shot, holding BELOPRZ. A big score was possible with the Z in hand, and MAVEN found BEZEL (O1, 57, OPR). Now Logan had to use the other TWS or else MAVEN could score using both of them.

Logan drew the QU, so he could match MAVEN score for score. Logan held BDDLOQU, and he played QUOD (1H, 50, BDL).

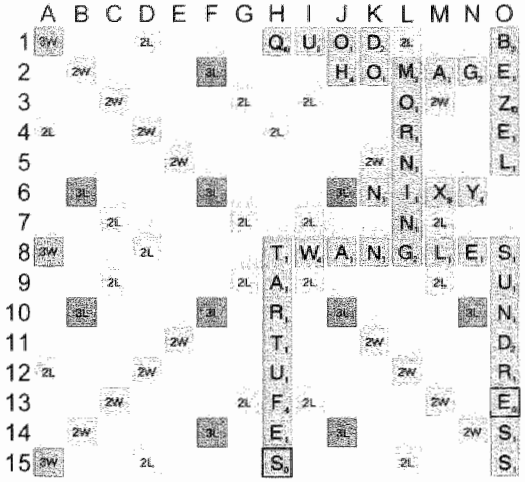
MAVEN's PORTION (11E, 36, J) was a nice double-double, and far superior to any other choice. It is the biggest score, and the J was set up nicely at 10F and 10J.

Logan played BODIES (14J, 30, AL), and frankly, I am surprised. During the game, Logan looked carefully at the triple-letter squares on the 10th row. Did he not see BORAL? BORAL (10F, 33, DEI) is better than BODIES from the perspective of both score and rack leave. The simulation margin is about 4 points.

MAVEN held AEEJMUY, and cashed its setup with JEU (10J, 53, AEMY). This balanced the rack leave better than alternative plays. For example, compare JAY (10J, 56, EEMU), which keeps the U among three vowels. JEU is about 5 points better.

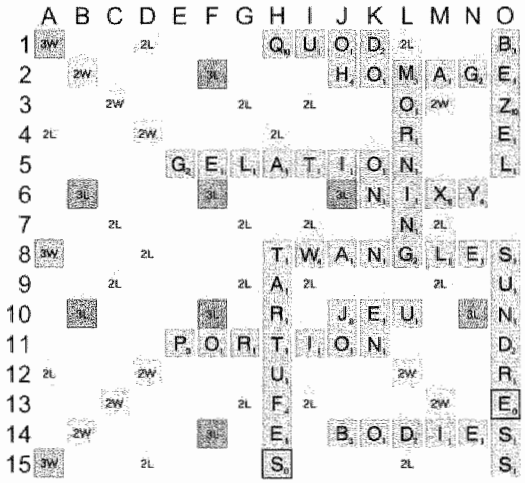
Logan nodded as if he thought a J play might be coming. No doubt, he was thinking back to BODIES. But he had no regrets as he slapped down a double-double bingo, out of AEGILOT: GELATION (5E, 90). LEGATION is also good. Logan preferred GELATION because the initial G is harder to overlap.

MAVEN has been out-bingoed, but has kept the game close with HOMAGE, PORTION, and JEU. MAVEN now took advantage of the TLS at 6F with FAY (6F, 37, AELM), which was best for both score and rack leave.



MAVEN:	I, J, N, O, O, P, R,	194
Logan's last:	QUOD (1H, 50)	207

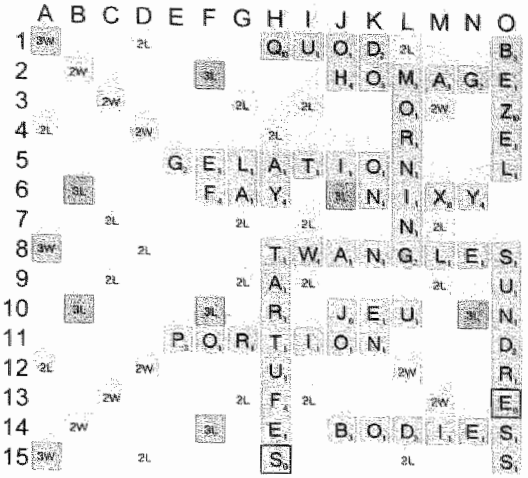
After Logan's QUOD (1H, 50)



MAVEN:	A, A, E, F, L, M, Y,	283
Logan's last:	GELATION (5E, 90)	327

After Logan's GELATION (5E, 90)

Logan chose HIP (4H, 31, AIOS). This is barely an error; he can play HIP at 13K for one point more. Logan might be attracted to the possibility of using a hook he has just created—HIP creates the word HAY vertically, and SHAY is legal. This creates an S hook for the S retained on his rack on a board that is otherwise bereft of hooks.



Logan: A, H, I, I, P, O, S, 327

MAVEN's last: FAY (6F, 37) 320

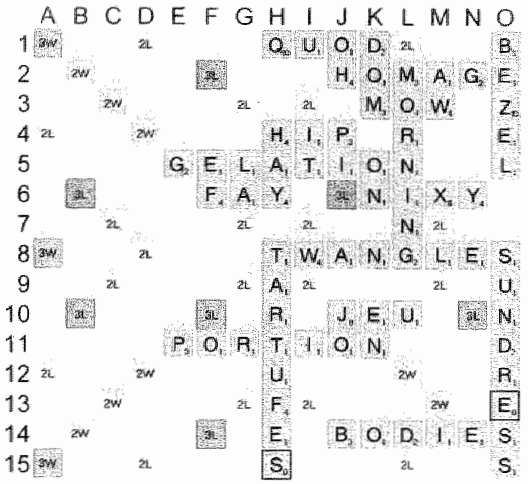
After MAVEN's FAY (6F, 37)

MAVEN maintains that PIA (4H, 23, HIOS) is best by a tiny amount over HIP (13K). My original notes to this game opined, "While this may be true from the perspective of point differential, I doubt that PIA will be as good at winning the game. HIP books 8 extra points that might not be regained with PIA."

Joel Wapnick reviewed my notes, detected the key positional issue favoring PIA, and objected, writing [51], "It sounds like you are using 'expert' logic here rather than MAVEN logic. PIA looks fine to me, because with the C/V distribution in the unseen tiles, it's better to have a 2-2 split than a 1-3 split. By far. Can you play these moves out to the end of the game and se which wins more often?" Well, I can, but it is easier to use MAVEN's winning percentage simulator, which uses a neural network to estimate winning percentage. It says that PIA wins more often, by 70.0% to 68.9%. Wapnick is right about the consonant / vowel split. The unseen tiles have 11 vowels and 14 consonants, so HIP's 3-vowel leave does hurt. I really ought to know better than to substitute my judgment for MAVEN's. But it is hard to improve MAVEN otherwise.

MAVEN held AELMRTW. MOW (3K, 32, AELRT) was the perfect dump for this situation. Only LOW played in the same spot comes close, but it would score 6 points less.

Logan's play was KIVA (12B, 26, EOS), but upon examination, there is no doubt that OAK (4D, 27, EIVS) is far superior. The attraction of KIVA is that it clears both K and V, leaving a clean rack to go with a good score. However, there is more to the situation.



Logan: A, E, I, K, O, S, V, 358
MAVEN's last: MOW (3K, 32) 352

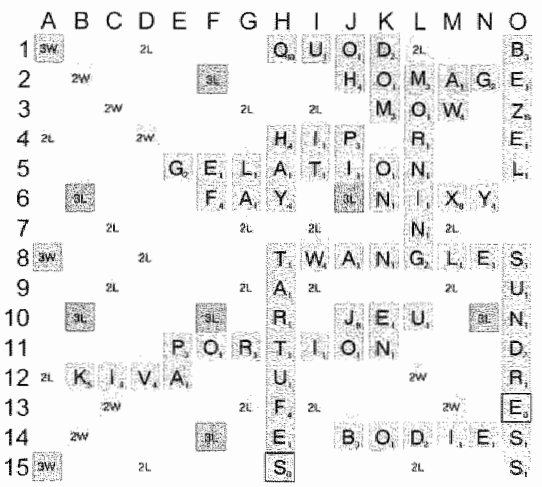
After MAVEN's MOW (3K, 32)

Consider the rack leaves at a dynamic level. KIVA keeps OES, which is sub-optimal because the O is not great and the vowel/consonant balance is adverse. OAK trades the O for an IV, which helps two ways. First, the vowel/consonant balance is ideal, and, second, the IVE group is useful.

The situation on the board is better after OAK, too. The big change is that the S-hook that Logan created over HAY moves to the C-column for SOAK, or along the 6th row for KEFS. This is better because the SHAY hook can only be used productively if the S is the last letter of a word of at least 6 letters. Such a word can be hard to assemble. The hook of SOAK can be used with the S in the middle of the word, and if Logan draws a word with the S at the end then he still has KEFS. The simulated point differential of OAK is 14.6 points higher than KIVA. The difference is surprisingly large.

MAVEN has cleared heavy tiles, and had a choice of two bingos: TREACLES (15A, 80) or RETACKLE (B7, 82). In terms of point differential, these moves are indistinguishable, with TREACLES having an edge of less than a point.

Since this is a PEG-9, MAVEN looked to the end of the game to find the move with the higher winning percentage. In that dimension, the difference is more pronounced. TREACLES wins 86.7% versus 78.8% for RETACKLE. The reason is that TREACLES left fewer bingo lines. Actually, the only one of any significance is the SHAY hook. RETACKLE leaves those plus bingos overlapping RETACKLE on the A-column and C-column.



MAVEN: A, C, E, E, L, R, T, 352
Logan's last: KIVA (12B, 26) 384
16 Unseen tiles: AACDEEEIIIORSTTV

After Logan's KIVA (12B, 26)

Logan's next move is one of the most instructive I have ever seen. I feel personally honored to have witnessed his selection process.

Logan began by checking that he had tracked the tiles correctly. He went over the board twice, because his first check did not agree with what he had tracked during the game. On the second check he found his error, and wrote down the unseen tiles AACIIRTTV on his scoresheet.

Logan: D E E E I O S 384
MAVEN's last: TREACLES (15A, 80) 432
9 Unseen tiles: AACIIRTTV

After MAVEN's TREACLES (15A, 80)

Next Logan calculated that his prospects from fishing a single tile were poor. Because he held EEE and two other vowels, he did not have a viable single-tile fish. There may be an odd bingo in there, but with MAVEN using its PEG-1 engine (and this was the ninth game of the match, so Logan had already experienced the PEG-1 engine's devastating accuracy), there was zero chance that any such bingo would land.

Having reached that conclusion, Logan realized that he had to fish two tiles in order to have any chance of bingoing. Unfortunately, a fish of two tiles would empty the bag, leaving MAVEN moving in an endgame. If Logan's tiles bingoed in only one spot then he was a goner, because MAVEN would block. Therefore, Logan needed to find a play that fished two tiles and maximized the probability that he would have bingos in two separated locations.

His next decision was to calculate which tiles to play. He had three real options. He could fish EE, EI, or EO. Which of these has the highest chance?

Logan's next action will never cease to amaze me. He enumerated all two-tile draws from the unseen tiles. There are 18 distinct draws, which he wrote on his scoresheet. Alongside each possible draw, he wrote the relative probability. For example, the draw AI occurs four times as often as AA, since there are two A's and two I's unseen. *He then calculated which of the draws made bingos in two separate spots for each of the two-tile fishes he was considering.*

Then he played OKE (B11, 7, DEEIS). Total elapsed time was about *two minutes*.

Simulation reveals that OKE is the best play by far. What's more, MAVEN does not generate this move within the top 10 plays, despite using four different move generators in the pre-endgame. Missing OKE is one of the biggest errors I have seen MAVEN make.

Joel Wapnick commented [51], "Adam's OKE was great. I would have played EKE without calculating, which apparently does almost as well." But Wapnick's next communication showed that he did the calculation, as he summarized the strength of OKE beautifully, "It has to be OKE so that there are two –S hooks." Precisely.

After Logan's OKE (B11, 7)

MAVEN knows its goose is cooked, and plays ACTA (10A, 18, IIT) to minimize the loss. MAVEN's move scores well and blocks REVISED (A5, 86). Then Logan bingoed out with DERIVES (3B, 84+6). Logan emerged with the victory by 481 to 450.

B.4 Wapnick-Cappelletto, WSC 2001

This game is a special treat. In the finals of the 2001 World Scrabble Championship (WSC), Joel Wapnick (1999 World Champion) faced Brian Cappelletto (2000 NSC Champion). This was the first game of the finals, a best-of-five match using the SOWPODS lexicon.

Wapnick's opening rack was AAIIOSU. Wapnick exchanged AIIOU, keeping AS, and the author could not be more proud of himself. In the old days this was an automatic exchange of 6, keeping only S. Thanks to MAVEN's persistent simulation results, experts now realize that they should keep additional one-point tiles, except for U, while keeping at least as many consonants as vowels.

Exchanging tiles is best by a mile. The only competition comes from the SOWPODS-only word AIA, but the best placement of AIA is about 9 points worse than exchanging. Wapnick kept the perfect pair, AS, which is slightly better than IS.

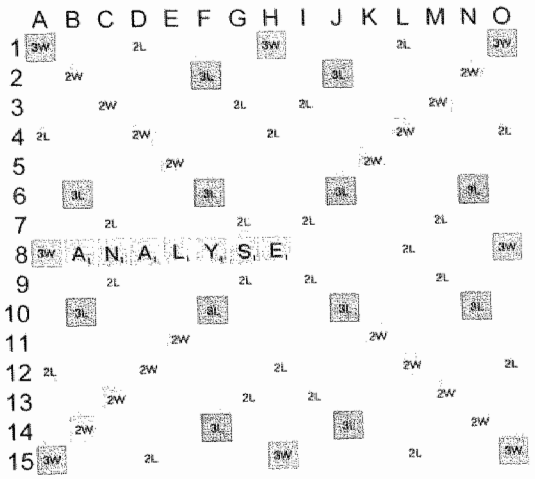
Cappelletto held EFLLOQT. Cappelletto exchanged FLOQ, keeping ELT. Again, these are the perfect tiles to keep. Cappelletto should consider keeping ELOT, but on balance, it is probably better to exchange the O. The O's Basic-1 value is –2.5, but improving from a 1-2 vowel-consonant distribution to 2-2 would only gain 0.5 points. Now if Cappelletto had an A instead then he should definitely keep it, and maybe he would even keep an I.

The real question is whether FLOTEL (8D, 26, Q) is better than exchanging. If it were not for an inference that Wapnick had kept good tiles then FLOTEL would clearly be

best, by about 4 points over exchanging. After Wapnick's exchange the balance between FLOTEL and exchanging is unclear. The difference between the plays is probably small.

Wapnick drew well, and held AAELNSY. Wapnick found ANALYSE, the only bingo. The question is where to place it. The highest scoring placement is at 8H, but that is the worst placement. We have previously mentioned the case of BRONZER, which wins more often at 8D than at 8H despite the extra 14 points scored by the 8H placement. ANALYSE is much worse at 8H because the 8H placement scores only 6 extra points, and ANALYSE takes back hooks of D, R, and S, making the chance that Cappelletto holds a hook roughly 85%. Wapnick chose the 8B placement, which is normal because it leaves no vowels adjacent to premium squares. MAVEN detects an advantage to the 8F placement, but it is not large. It is time for a diagram.

Cappelletto played IRATELY (I7, 77). Cappelletto had 12 bingos, and he correctly chose the highest scoring. The second-best play was INTERLAY (C7, 76), and he had IRATELY, REALITY, and TEARILY in several spots.



After Wapnick's ANALYSE (8B, 72)

At this point Wapnick challenged. The WSC adopts special rules for adjudicating challenges. This is necessary because the North American game uses *penalty challenges*, in which a challenge of a valid word costs the challenger a turn, whereas the United Kingdom uses *free challenges*, where challenging a valid word does not cost a turn. Since the WSC uses a dictionary (SOWPODS) that many participants do not regularly use, the WSC uses the free challenge rule. Alas, the history of the event shows that with nothing to lose from a challenge, players routinely challenge every unfamiliar word. This is fine for the players, but the tournament staff has to run a marathon to keep up with the workload. Accordingly, in 2001 a failed challenge incurred a 5-point penalty.

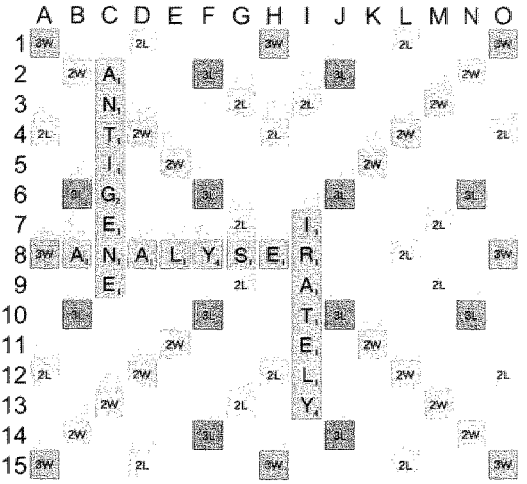
Five points is a small risk, especially when challenging a 77-point play, so Wapnick chanced it, wondering if ANALYSER was really in the OSW. Wapnick would have been certain that ANALYSER was not in TWL98. Cappelletto pocketed 5 points because the play is acceptable.

Wapnick held AEEGINT after a great draw from the bag, and had a choice of bingos. He had ANTIGENE from C2 and C7, and GALENITE / GELATINE / LEGATINE in several places, and SAGENITE. ANTIGENE from C2 and C7 simulate best. MAVEN's simulations show that ANTIGENE from C2, as played by Wapnick, is the best move by a hair over ANTIGENE (C7).

Note that Wapnick followed his challenge with a bingo. For this reason alone, if Wapnick risked losing his turn for challenging then he probably would not have challenged.

Cappelletto's rack screamed for LUVS, and, in fact, the two best moves are LUVS in two places. Cappelletto chose LUVS (J4, 17, DSU), which is best. Second best is LUVS at E8, scoring only 14.

Wapnick followed with POLTROON (E6, 68), his third consecutive bingo. However, Wapnick dropped 10 points by missing PATROONS (B7, 78). The cost of this error in terms of point differential is about 7 points, and the cost in winning percentage is about 1%. The cost in winning percentage would have been greater had Wapnick not been leading to begin with.



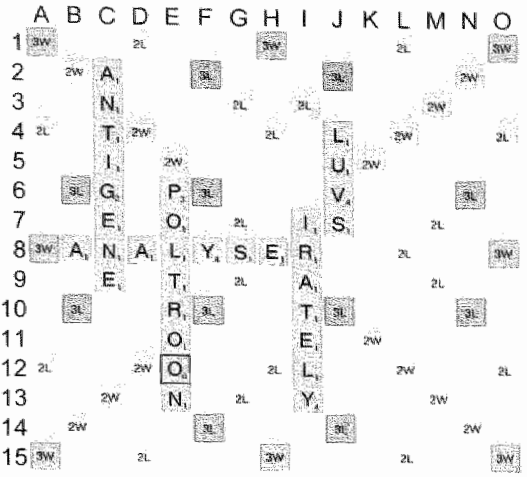
Cappelletto: D, L, S, S, U, U, V, 82

Cappelletto: ANTIGENE (C2, 72) 144

After Cappelletto's ANTIGENE (C2, 72)

Wapnick might have missed PATROONS, but he is unlikely to have missed POLTROON (4H, 76), since he found POLTROON at E6. We conclude from this that Wapnick believed, over the board, that the triple-triple opening was a big enough threat to his lead to outweigh 8 extra points. I would not have passed up the 8 points, but simulations show that the two POLTROONS have equal winning chances. Who am I to argue?

Cappelletto replied with INDUSIA (14A, 78) his highest scoring and best play. He had many other bingos. It is likely that Cappelletto homed in on INDUSIA by reasoning that if there were a bingo hooking POLTROON then it would be the biggest because the position does not offer triple word or double-double opportunities.



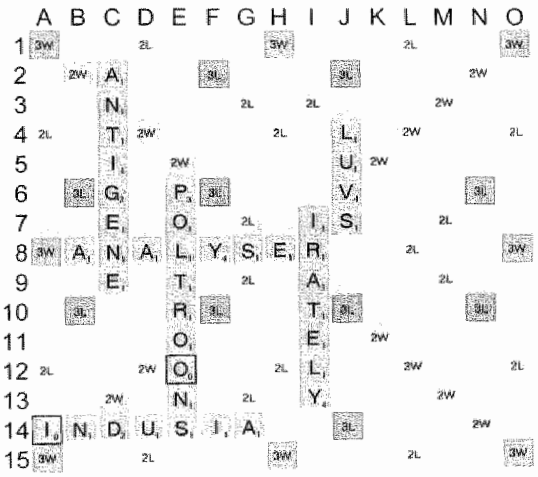
Cappelletto: A, D, I, N, S, U, _ 99
Wapnick's last: POLTROON (E6, 68) 212

After Wapnick's POLTROON (E6, 68)

Note that Cappelletto could have used the blank as either I, but chose to make the blank stand for the first I. There is a good reason for this: the other I stands on a triple-letter square, so Cappelletto gains more points by calling his blank as he did.

Wapnick's BAJU (15G, 43, DGV) is best, but not by as much as you might think. In SOWPODS, the tile V is a genuine millstone, so GUV (15G, 24, ABDJ) is only a 4-point sacrifice, and only costs 1% in winning percentage.

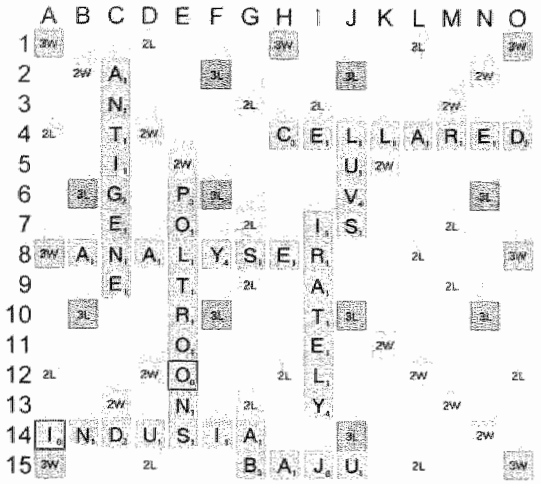
With the rack ACDEELR, Cappelletto evened the game with CELLARED (4H, 82), again his highest scoring bingo and the best play. He has alternatives (RECALLED (4H, 77)) that do not open triple word lines. However, CELLARED is best despite such openings. Actually, in this position the triple word openings cost nothing. Triple word lines are less significant, in general, on a wide-open board. Additionally, CELLARED opens two lines that balance each other. The triple-triple consideration is not important to Cappelletto, of course.



Wapnick: A, B, D, G, J, U, V, 212
Cappelletto's last: INDUSIA (14A, 77) 177

After Cappelletto's INDUSIA (14A, 77)

Wapnick's reply was FOH (3L, 32, DEGV), on top of CELLARED, but HOUF (D12, 38, DEGV) is stronger. I wonder why Wapnick missed HOUF. I can only speculate. What follows is an example of how humans can go wrong. It is not meant to suggest how Wapnick went wrong, because great players do not necessarily have the same blind spots as other humans. Wapnick wrote [51], "It's no great mystery why I didn't play this move. I didn't see it. I know the word." While this comment obviates some of the interest in speculating, it is an interesting psychological question nonetheless, even though we know what happened to Wapnick in the actual game. Many errors in move generation originate from the goal-oriented nature of human move generation. For instance, in this game, a human that had played 3 consecutive bingos might play a move under the "retained image" of his previous lead. He might make a move as if he still had a lead, even though his opponent has just erased it. This would cause a preference for plays parallel to CELLARED, since such plays block many bingo lines. FOH also interferes with the triple word line. A human would have tried to find a playing using the O-column, but found nothing better then DEFOG (O4, 30, DHV). The focus on moves in the upper left would reduce attention on moves elsewhere.



Wapnick: D, E, F, G, H, O, V, 255

Cappelletto's last: CELLARED (4H, 82) 259

After Cappelletto's CELLARED (4H, 82)

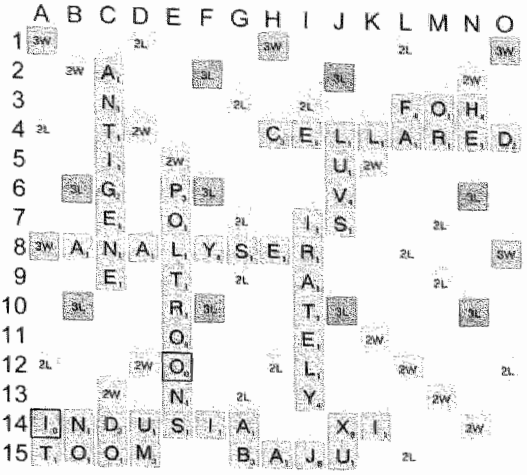
From IIMOORT, Cappelletto played TOOM (15A, 42, IIR), which is best despite the dreadful II. Other plays (E.g., MOIT (B2, 16)) score too little.

Wapnick held DEGIIVX. Wapnick's XI (14J, 50, DEGIV) was an automatic play. GIVED (14J, 35, IX) is next best.

Cappelletto played MILKIER (12G, 28, R), but missed DIKIER (O4, 33, MR). DIKIER's simulation results are 3.5 points higher than MILKIER's. MILKIER reduced Cappelletto's winning chances by about 4%. Also better than MILKIER is ERICK (H1, 33, IMR).

Wapnick held DEEGIOV. Wapnick's VOICED (H1, 36, EG) is best. Second best is DEVOID (O4, 33, EG).

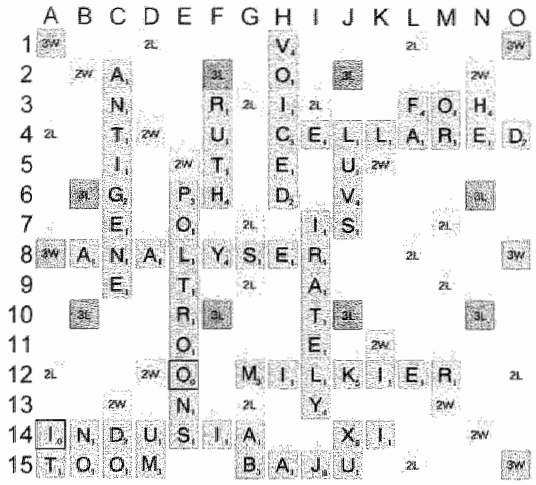
Then Cappelletto held AHNRTUZ. Cappelletto played RUTH (F3, 30, ANZ). RUTH uses the right spot, but Cappelletto missed the best play, THUYA (F5, 38, NRZ). The error cost 3.5 points, and about 3% of his winning chances. Joel Wapnick's simulations [51] put TARZAN (B7, 39, HU) on top of THUYA, but I think that result is probably wrong. Not by much, though, since my simulations show TARZAN as 3rd best, only a point behind THUYA. The difference is probably due to either the number of iterations (i.e., statistical noise) or to rack tuning. I would bet on the latter. Wapnick's version of MAVEN would have the SOWPODS lexicon but rack parameters tuned to OSPD2. The second best play is HAND (O1, 35), which uses a hotspot that both players appear to have overlooked: FOH takes a back N hook to make FOHN. The hidden strength of THUYA is that it keeps an N for O1 plays such as ZANDER and ZONDA.



After Wapnick's XI (14J, 50)

As the late game commences, positional considerations become subtler and more important. Some of the differences between plays are hard to detect. Additionally, as the average scores drop, hotspots that were previously insignificant gain value. Another impact of resource depletion is that the players may increase fighting over hotspots. In this environment, the impact of overlooking a hotspot may cause a series of errors. When both players overlook a spot, as in this game both players overlooked the significance of the –N hook to FOH, then the players may make the best decisions from the perspective of the information they have available to them, but the annotator labels every move as an error.

Wapnick played GEE (11K, 18, BCEG), which is a good move. However, EGG (D10, 19, BCEE) is better by a stupendous amount. MAVEN rates EGG as 10 points better, with a winning percentage advantage of 58% against 48% Why the big difference? The whole truth is that I am not sure. Table 0-4 shows move-by-move average scores. MAVEN is simulating to the end of the game, so the variations encompass more than two plies. The table shows four plies explicitly, and then gives the rest of the moves (some of which might not even occur in every game) as "Future."



After Cappelletto's RUTH (F3, 30)

Move	Score	Opp1	Our1	Opp2	Our2	Future	Total
EGG	19	38.9	35.5	40.4	27.9	10.1	-7.0
GEE	18	34.2	32.8	38.3	22.9	17.8	-16.6

Table 0-4 Simulation of EGG and GEE

The table shows that the two-ply values of the plays are nearly equal, but EGG has a large edge in the Our2 and Future categories. I am not sure why there is a difference. I cannot trace the difference to any specific issue. EGG does seem like a better play on general principles, since it scores one point more and keeps a better vowel / consonant balance than GEE, and the tradeoff between keeping a G and keeping a duplicate E is basically neutral. If I had to summarize the difference, it would be that EGG shows better timing heading into the endgame. What gives me pause is that the players will manipulate endgame timing on every remaining turn, so it is not clear that manipulations at this distance from the horizon are meaningful. Some of EGG's advantage could be an artifact of selecting moves in the simulation using the static analyzer, which is insufficiently sensitive to the endgame timing issues of *this* position.

These misgivings notwithstanding, I am certain that EGG is a better play because it has better general characteristics, and the simulation results are at least as good as GEE's. The real question is whether an altogether different move is best. For example, ABCEE (B8, 19, EGG) has the second highest results. Is ABCEE better than EGG? There is always work for another day…

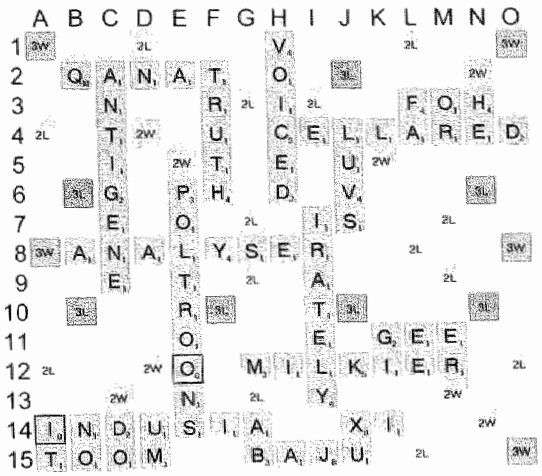
Cappelletto played the natural move QANAT (2B, 42, AOZ), which is best. Another move was ZONDA (O1, 55, QAT), which plays a significant role in the tactics of this game. QANAT is better because the Z has many productive places to play. The difference in value between Q and Z is apparent in the Our1 column of Table 0-5 of move-by-move averages.

Move	Score	Opp1	Our1	Opp2	Future	Total
QANAT	42	33.7	48.7	24.3	1.0	33.7
ZONDA	55	34.9	31.8	29.1	5.7	28.5

Table 0-5 Simulation of QANAT and ZONDA

This table shows that the transition to the endgame is again a significant event, boosting ZONDA by 4.7 points, as shown by the Future column of the table.

After QANAT, Wapnick played WHEEP (N2, 38, BCEG), which is the second best move. Stronger is CHEEP in the same spot (see Table 0-6). The difference is that big plays from O1 are solidly blocked. WHEEP allows many such plays to overlap cleanly. CHEEP scores 2 points less than WHEEP, but has a point differential 1.5 point higher, mainly because of the opponent's next play. This small difference is magnified by the fact that plays from O1 can be game-changing. WHEEP cost Wapnick about 5% in winning percentage.



After Cappelletto's QANAT (2B, 42)

Move	Score	Opp	Future	Total
CHEEP	36	40.2	14.6	10.5
WHEEP	38	43.9	14.7	8.8

Table 0-6 Results of CHEEP vs WHEEP

Cappelletto played FRIZERS (M7, 48, AO). This is the biggest score, but it goes too deep into the endgame. There are only 2 tiles in the bag, so Cappelletto will have 4 tiles after his draw. If he had a good chance of going out in one turn, then FRIZERS would be a good move, but going out in one is actually unlikely. Because Wapnick moves first in the endgame, Cappelletto needs *two* plays that go out in order to have a forced out in one. Cappelletto is better off keeping six tiles, which ZATI (4A, 46, FORS) accomplishes. ZATI also keeps an F and an S for scoring muscle. ZATI has a point differential only 2 points higher than FRIZERS, but the edge in winning percentage is 12%.

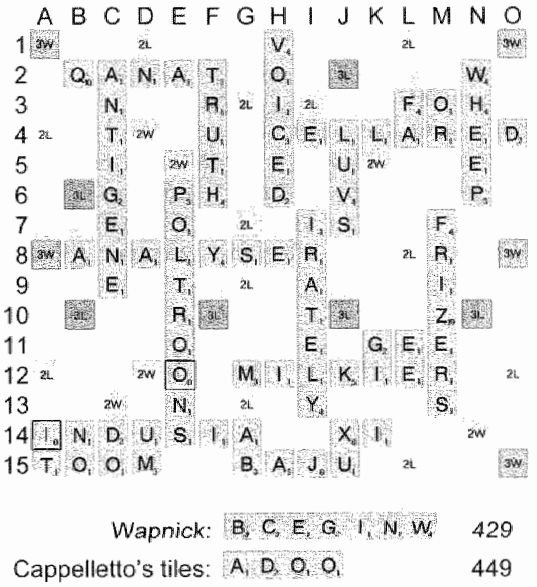
FIZ (13M, 42, AORS) is almost as good as ZATI. FIZ leaves an endgame with 7 tiles versus 6. It also creates a triple-word line through the open Z. This opening costs points on average, of course. But when things break right for Cappelletto then the opening can

win the game. MAVEN estimates that FIZ has a point differential 7.5 points lower than ZATI, but the winning percentage is only 2% lower.

The endgame starts with Wapnick trailing by 20 points. Wapnick's next was CREW (8L, 36, BGIN), an error that should have cost Wapnick the game. Best was CRIB (8L, 33, EGNW), because the tiles EGNW retained a big threat: WEND (O1, 34, G). This threat would force Cappelletto to block with DO (O1, 14, AO). After Wapnick's WANG (B7, 19, E) and Cappelletto's ZOA (10M, 14+2), Wapnick would emerge with a 22-point gain and a 2-point win. Commentary during the game stated [51], "Wapnick played a terrific endgame play of CREW for 36, setting up an out play: BINGO." The commentary is wrong; CREW does not set up an out play. CREW *threatens* an out play, which is not the same at all because Cappelletto will block. As a *move*, WEND is smaller than an out play, but it is just as big a *threat* because it, too, will be blocked. Wapnick benefits because Cappelletto's block of WEND is smaller then his block of BINGO.

In the game, Cappelletto followed CREW with ADO (D10, 15, O), an error that threw the game back to Wapnick. Cappelletto must block the plays BINGO (12A, 20) and BINGO (11A, 8), which would otherwise go out. However, Cappelletto should have blocked with DOUM (D12, 20, AO), which leaves a larger out-in-two. After DOUM, Wapnick's BANG (B7, 17, I) and Cappelletto's ZOA (10M, 14+2) leaves Cappelletto with a 19-point gain and a 3-point win. After Cappelletto's ADO, Wapnick played BANG (B7, 17, I), and Cappelletto played ZO (10M, 13+2), which netted only 13, for a 3-point loss for Cappelletto.

Thus, the game ended in a 481-478 victory for Wapnick, but Cappelletto swept the next three games to win the World Championship.



After Cappelletto's FRIZERS (M7, 48)

Wapnick's Error	Points	Equity
Bad challenge	5	2%
POLTROON	10	1%
FOH	7.5	3%
GEE	10	10%
WHEEP	3.5	5%
CREW	5	100%

Cappelletto's Error	Points	Equity
MILKIER	5	4%
RUTH	6	6%
FRIZERS	2	12%
ADO	6	100%

Table 0-7 Error Analysis of Wapnick-Cappelletto

In my opinion, the play of this game was exceptionally accurate. Table 0-7 shows the error analysis. The players never missed a bingo, and generally played either the best or second best move. It is remarkable that the players accomplished this feat while playing in SOWPODS, which is a more difficult vocabulary than TWL98, and is not the habitual vocabulary of either player.

Appendix C – Historical Timeline

1983: First program, in PL/1 on an OS/370.

1986: First MAVEN, in C on a DEC VAX 11/780. Competed in first tournament, scoring 8-2.

1987: Major dictionary overhaul.

1987: MAVEN development moved to Apple Macintosh. Still in C. Competed in second tournament, scoring 5-0.

1988: Switched to Appel-Jacobson move generator.

1988: First successful endgame player.

1988: Competed in third tournament, scoring 7-3.

1989: First endgame player that incorporated B*.

1989: First applied simulation for research purposes.

1990: First pre-endgame player.

1993: Ported to Windows platform, using Borland C++.

1995: MAVEN incorporated into the Scrabble CD-ROM from Hasbro Interactive.

1997: First used simulation for choosing moves in real time. MAVEN lost to Adam Logan by 0-2.

1998: MAVEN defeated Joel Sherman and Matt Graham by 6-3.

1998: Rewrote in Visual C++. Major debugging of pre-endgame player. Validated endgame player resulting in a few minor bug fixes. Defeated Adam Logan by 9-5. Fixed some bugs uncovered by analyzing the match.

1998: Contributed to the Second Edition of the Scrabble CD-ROM.

1999: Added support for non-English languages.

1999: Implemented an endgame evaluation function for use within the simulator.

2000: Implemented a neural network that evaluates winning percentage within the simulator.

2001: Began this thesis.

2002: Implemented a scalable simulation controller, removing many limitations.

2002: Create a prototype inference engine.

2002: Presented this thesis.

Index

Summary

In this thesis, we describe the history, techniques, and results of the MAVEN Scrabble engine. MAVEN started out as a research project to explore the limits of computer Scrabble playing ability. It developed into an exceptionally robust engine that dominated the human champions of the game and contributed to our understanding of the game.

Chapter 2 describes the game of Scrabble, concentrating on the nature of the game and the skills needed to perform at a high level.

Chapter 3 sketches the history of computer Scrabble from the author's point of view. The literature contains few papers about computer Scrabble; this thesis presents much of this material for the first time.

Chapter 4 covers the purely algorithmic task of move generation. If you want to play Scrabble at a high level, then move generation must be exhaustive and fast. Ingenious algorithms and data structures solve these problems.

Chapters 5 and 6 describe how to evaluate moves. MAVEN was programmed to develop a positional theory through self-play. Chapter 5 shows how to evaluate changes to the rack. Chapter 6 discusses how to evaluate changes to the board.

Chapters 7, 8, and 9 show how the game evolves in stages. Chapter 7 characterizes the first stage, the Early Game. The final stage of the game the Endgame, which Chapter 8 covers in detail. Chapter 9 deals with the Pre-Endgame, which is the phase between the two. We will tackle the Pre-Endgame last because it has aspects of both the Early Game and Endgame.

Chapter 10 describes Simulation, a technique that revolutionized computer Scrabble. The technique is an implementation of Monte Carlo search of the state space. The application of Monte Carlo search to Scrabble is successful, so it is worth careful study.

Chapter 11 describes opportunities for improvements in simulation. Opportunities arise because simulation involves many policy choices for how to draw racks, play out variations, evaluate endpoints, and control the search. The implementation in MAVEN is but one example. Though that implementation has demonstrated practical success, there are still open questions.

Chapter 12 presents MAVEN's competitive results. MAVEN has played in three tournaments and three arranged matches. These results validate the quality of MAVEN's implementation.

Chapter 13 describes opportunities for further investigation. While research on MAVEN has been a resounding success, there remain areas where further investigations may be fruitful.

Chapter 14 summarizes research results.

Appendix A gives the rules of the game.

Appendix B presents three annotated games. MAVEN software validated the annotations.

Appendix C gives the historical timeline of MAVEN's development.

Samenvatting

In dit proefschrift worden de geschiedenis, de technieken en de resultaten van het Scrabble-programma MAVEN beschreven. MAVEN is begonnen als een onderzoeksproject naar de mogelijkheden en beperkingen van computers om Scrabble te spelen. Het heeft zich ontwikkeld tot een zeer sterk en robuust programma dat de menselijke Scrabble-kampioenen heeft verslagen. Tevens heeft het bijgedragen aan een beter begrip van het spel.

In hoofdstuk 2 wordt de aard van het Scrabble-spel beschreven en de vaardigheden die nodig zijn om op een hoog niveau te presteren.

Hoofdstuk 3 geeft een beeld van de geschiedenis van computer Scrabble vanuit de auteur gezien. Er zijn in de literatuur weinig artikelen over computer Scrabble te vinden en in dit proefschrift wordt veel van het materiaal voor de eerste keer gepubliceerd.

Hoofdstuk 4 behandelt het zuiver algoritmische probleem van het genereren van zetten. Wanneer men Scrabble op een hoog niveau wil spelen dan moet het genereren van zetten niet alleen uitputtend maar ook snel zijn. Er bestaan ingenieuze algoritmen en data-structuren om deze problemen op te lossen.

De hoofdstukken 5 en 6 geven een methode aan om zetten te evalueren. Er is een programma ontwikkeld waarmee MAVEN tegen zichzelf kan spelen; dit heeft geleid tot een positionele theorie. In hoofdstuk 5 wordt aangegeven hoe men veranderingen op het rek evalueert. Hoofdstuk 6 gaat in op de evaluatie van de veranderingen op het bord.

De hoofdstukken 7, 8, en 9 beschrijven hoe het spel zich in fasen ontwikkelt. Hoofdstuk 7 gaat in op de eerste fase, de opening en het beginnende middenspel. De laatste fase van het spel, het eindspel, wordt gedetailleerd behandeld in hoofdstuk 8. Hoofdstuk 9 gaat in op het "pre-eindspel", dat is de fase is tussen opening en eindspel. We behandelen het pre-eindspel als laatste omdat het aspecten van beide fasen behelst.

Hoofdstuk 10 geeft een beschrijving van de simulaties. Deze techniek heeft voor een doorbraak in het computer Scrabble gezorgd. De techniek is een toepassing van het Monte Carlo zoeken in de toestandsruimte. Deze toepassing is voor Scrabble zeer succesvol, hetgeen een zorgvuldig onderzoek rechtvaardigt.

Hoofdstuk 11 bespreekt diverse mogelijkheden voor verbetering van de simulatie. Deze mogelijke verbeteringen betreffen veelal strategische keuzen over het trekken van letters, het uitspelen van varianten, het evalueren van eindtoestanden, en het besturen van het zoekproces. De verwezenlijking hiervan in MAVEN is slechts een voorbeeld. Hoewel de daadwerkelijke implementatie kan bogen op een groot aantal successen, zijn er toch ook nog open vragen.

Hoofdstuk 12 toont de wedstrijdresultaten van MAVEN. Het programma heeft deelgenomen aan diverse toernooien en officieel georganiseerde matches. De resultaten bevestigen de kwaliteit van het programma MAVEN.

In hoofdstuk 13 worden de mogelijkheden voor verder onderzoek besproken. Hoewel het MAVEN-onderzoek klinkende successen heeft opgeleverd, blijven er gebieden over waarin verder onderzoek tot vruchtbare resultaten kan leiden.

Hoofdstuk 14 geeft een samenvatting van de onderzoeksresultaten.

Appendix A beschrijft de regels van het spel.

Appendix B vermeldt drie geannoteerde wedstrijden. MAVEN-software heeft deze annotaties gevalideerd.

Appendix C geeft de historische tijdlijn van de ontwikkeling van MAVEN.

SIKS Dissertatiereeks

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Withdrawn

2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel (KUB)
Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

Curriculum Vitae

Brian Sheppard was born in New York City on 4 June 1962. He attended New York City's public schools. He graduated from the Bronx High School of Science in 1979.

In high school, Brian competed in local and national mathematics competitions. He succeeded in winning the individual championships of New York City and New York State in 1979. He was co-captain of New York City's Math Team, which represented the city at regional and national competitions.

Brian also wrote and presented mathematical papers. He was a Westinghouse (now Intel) Science Talent Search Finalist in 1979 for his paper on a sub-domain of the integers that lacks unique prime factorization.

Brian went to Harvard College. In 1984, after many harrowing experiences, he graduated with a BA in mathematics and a minor in Ultimate Frisbee. He twice managed to earn honors on the William Lowell Putnam mathematics competition, despite the fact that the exam is administered in the morning.

Brian has a lifelong fascination with games, and especially with programming them. At Harvard, he did an independent study of Othello programming under Professor Reif. While away from Harvard for a year spent working at IBM's Yorktown Research Center, Brian discovered that research into Scrabble programming was in its infancy, and that discovery changed his life forever.