CS-TR-0002 May 2006

AN OPTIMAL STRATEGY FOR YAHTZEE

James Glenn

Loyola College in Maryland
Computer Science Department
4501 N. Charles St
Baltimore, MD 21210-2699

Abstract

Solitaire Yahtzee is a complicated game in that the best strategy is not obvious, as it is for simple games like tic-tac-toe, however it is not so complicated that computing the optimal strategy is infeasible. The elementary combinatoric and graph theoretic techniques used to compute the optimal strategy for Yahtzee are described. Once the strategy has been computed, the same techniques can be used to perform statistical analysis of it and other non-optimal strategies. Some results of this analysis are presented.

Keywords: Retrograde Analysis, Computer Games, Computer Strategies

The work for this paper was performed while the author was at the University of Maryland, College Park, and later at Lafayette College, Easton PA.

An Optimal Strategy for Yahtzee

James Glenn
Department of Computer Science
Loyola College in Maryland
4501 N. Charles St.
Baltimore, MD 21210
email: `jglenn@cs.loyola.edu`

May 10, 2006

Abstract

Solitaire Yahtzee is a complicated game in that the best strategy is not obvious, as it is for simple games like tic-tac-toe, however it is not so complicated that computing the optimal strategy is infeasible. The elementary combinatoric and graph theoretic techniques used to compute the optimal strategy for Yahtzee are described. Once the strategy has been computed, the same techniques can be used to perform statistical analysis of it and other non-optimal strategies. Some results of this analysis are presented.

1 Introduction

Many games, for example tic-tac-toe, are simple enough that most young children can master the strategy. Other games, for example checkers and chess, have such large state spaces that it is inconceivable that a computer could compute the optimal strategy; for these games we must resort to complex heuristics. In between stands Yahtzee, a game for which the optimal strategy is not immediately obvious, but can be computed given even modest computational power. Later we give some results of this computation; similar work has been undertaken by Tom Verhoeff [2] (happily, our computations ended in the same results).

Yahtzee is a game of skill and chance marketed by Hasbro (a similar game called "Yacht" is described in *Hoyle's Rules of Games* [1]). Yahtzee is played with five six-sided dice. The goal is to roll the dice to make hands resembling poker hands. Different hands are worth different amounts of points. The goal of a two-player game is to end up with more points than one's opponent. In this paper we describe how to compute the optimal strategy for the solitaire game in which the goal is to maximize one's long term average score.

2 Rules

A game of Yahtzee lasts thirteen rounds. Each round begins with the player rolling five six-sided dice. After the initial roll, the player may choose to reroll any of the dice. After the first reroll, the player is given a second chance to reroll as many of the dice as desired. After exhausting the rerolls (or choosing to keep all five dice), the player must classify his five dice in one of thirteen categories. Each category may be chosen only once per game, and once a roll has been placed in one category it may not be changed. Points are awarded based on the roll and the category chosen.

The thirteen categories and the corresponding scoring rules are as follows:

- **ones** scores one point for each die showing one pip;
- **twos, threes, fours, fives, sixes** similar to ones;
- **three of a kind** scores the total pips showing on all the dice if at least three show the same number of pips, zero otherwise;
- **four of a kind** similar to three of a kind;
- **full house** scores 25 points if the roll has three dice showing one rank and two dice showing another, 0 otherwise;
- **small straight** scores 30 points if four dice show consecutive values, 0 otherwise
- **large straight** scores 40 points if all five dice show consecutive value, 0 otherwise
- **chance** scores the total of all dice (with no additional restrictions);
- **five of a kind** (or "yacht" or "Yahtzee") scores 50 if all five dice show the same number of pips, 0 otherwise.

Additionally, bonus points are awarded in certain circumstances. Players who score at least 63 points in ones through sixes (the "upper categories"), earn an additional 35 point "upper bonus". There is a variation of the rules that allow players who score 50 in Yahtzee to earn a "Yahtzee Bonus" of 100 points for each subsequent round that ends with all five dice matching. The player must still score the roll in some other category, and in fact in some cases can use a second Yahtzee to earn the full score in small straight, large straight, or full house (a "Yahtzee Joker").

The strategy for Yahtzee is more complex than the strategy for poker in this sense: while the choices you make in one poker hand do no affect your chances of getting a good hand in the next, your choices in one round of Yahtzee do affect your chances for scoring well in the next round: if, for example, you choose to score 1 1 1 2 3 as three of a kind (8 points), you may not later score 4 5 6 6 6 there (27 points).

We will determine how to figure out the best strategy for Yahtzee. Our strategy will tell us which dice to reroll when we have the opportunity, and which category to put our roll in at the end of each round.

3 Method

To figure out the best strategy, we envision the game as a graph. Vertices represent states of the game. At some states the player has a choice as to which state to go to next. From those states there will be edges to the subsequent states. In our pictures, we join the edges with an arc.

At other states some random event occurs. The subsequent states will again be indicated by outward edges in the graph. This time, we put weights on the edges indicating the probability that we end up in the destination state.

To illustrate, consider a simpler game. This game is played with two fair coins. The player flips both coins simultaneously. After the first flip, the player may flip either, none, or both of the coins again. After reflipping, the player wins 1 dollar if both coins came up heads and loses 1 dollar if one coin shows heads and one shows tails; no money changes hands in the remaining case.

There is a vertex in the graph for each state the game can be in. There is an edge from state u to state v if there is an action that will move the game from state u to state v. In this coin tossing game, there are four kinds of states: one state with indegree zero representing the beginning of the state; three states representing the three possibilities for the outcome of the first flip; six states representing the possibilities for which coins to keep; and three states with outdegree zero representing the three possibilities for the outcome of the second flip. When drawing the graph, it is convenient to group each kind of state into a column. Edges from the initial state to states in the second column and from the third column to the final states represent the outcome of a random event (flipping the coins); these edges are labelled with the probability of the outcome. Edges from the second column to the third column represent choices the player can make; in the picture the edges representing all the choices a player has at a given state are joined by an arc.

For example, in the initial state the player will flip two coins. There is a $\frac{1}{4}$ chance that the toss will result in two tails, a $\frac{1}{2}$ chance that the toss yields a head and a tail, and a $\frac{1}{4}$ chance that both coins land heads up. Having tossed two tails, the player has the choice of reflipping the tail, reflipping the head, or keeping both coins. If the player chooses to reflip the tail, there is a $\frac{1}{2}$ chance he will end the game with two heads and a $\frac{1}{2}$ chance he will end the game with one head and one tail. The complete graph for this game is in Figure 1.

Given the graph, we can compute the strategy that will maximize average winnings, and figure out the average winnings if the player follows that strategy. To do so, we will assign a value $VAL(u)$ to every state u in the graph. Since the graph has no cycles, we can proceed in order of a reverse topological sort: all vertices with no outgoing edges are final states of the game and their values can be computed immediately from the table of payouts given the the game

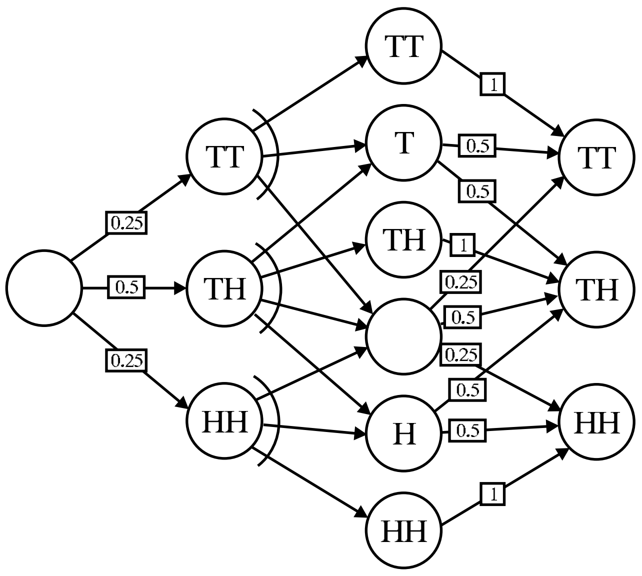

Figure 1: State diagram for a coin tossing game.

description; the values for other vertices can be computed once the values have been computed for all vertices to which they have outgoing edges. There are two kinds of such vertices: vertices where a random event occurs and vertices where the player makes a choice. The value of a state u in which a random event occurs is

$$\sum_{(u,v)\in E} w(u,v) \cdot VAL(v).$$

The value of a state u in which the player has a choice as to what state to move to next is

$$\max_{(u,v)\in E} VAL(v)$$

and the best choice to make is to move to the state that realized the maximum value.

Most of the strategy for this game is obvious: the player should keep any heads obtained by the first flip. But when the first flip yields two tails there is a decision to be made: either stand pat and take the draw or reflip both coins in the hopes of winning (the option of reflipping only one coin is clearly inferior because there is no chance of winning but there is a chance of losing). From the computed values for each state in Figure 2, we see that, having flipped two tails, the player should stand pat and take the draw rather trying to go for a win. A little algebra shows that, keeping the other outcomes the same, if the payout for two heads is higher than two then the best strategy is to reflip both coins.

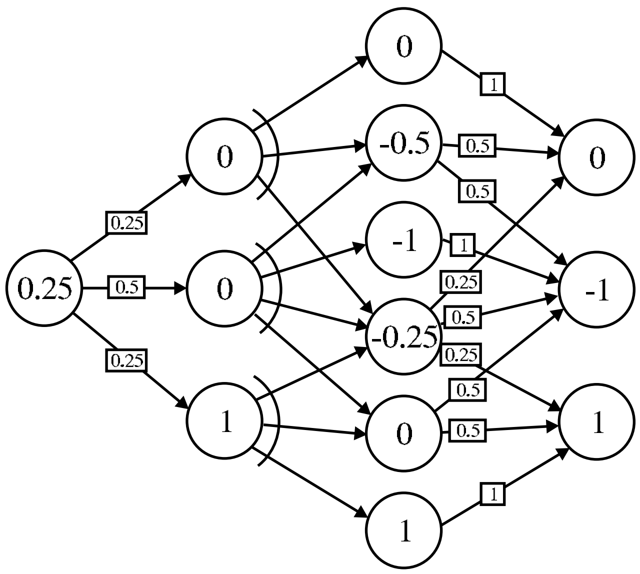

Figure 2: Values of states for a coin tossing game.

4 Yahtzee State Graph

Computing the optimal strategy for a complete 13-round game of Yahtzee is similar to computing the optimal strategy for the coin tossing game. Once we figure out all the states of the game and all the possible actions and their consequences, we can work backwards from the final states (where all categories are used) toward the initial state and figure out what the ideal strategy is and what our average final score will be when following that strategy.

4.1 Structure of the graph

There will be several different kinds of states in our graph to represent the steps in each turn just as there were in the graph for the coin tossing game. The graph for the coin tossing game has states where the player chooses which coins to keep; the graph for Yahtzee has states where the player chooses which dice to keep or what category to score a roll in. The graph for the coin tossing game has states where the player flips coins; the graph for Yahtzee has states where the player rolls the dice.

In addition, states in the Yahtzee graph must keep track of some of the history of the game, since Yahtzee strategy depends on what has happened in previous turns. That was not a concern in the coin tossing game because there was only one round.

We could keep track of *all* the information about the state of the game; that is, for each category what, if anything, has been scored in it. There are seven possibilities for what has been scored in each of categories ones through sixes: either the category is empty, or is has been used for zero, one, two, three, four, or five of the given dice. There are 28 possibilities for chance, three of a kind, and four of a kind (unused, zero, or anywhere from 5 to 30 points). There are three possibilities for each of the other categories (unused, zero, or nonzero). So the total number of ways to mark the scorecard is $7^6 \cdot 28^3 \cdot 3^4 = 209{,}193{,}098{,}688$. Recording information about that many states uses a prohibitive amount of memory, not to mention time. Obviously we must be more careful.

Observation 4.1 *The best strategy does not depend on what has been scored in each category (with one caveat), but rather on what categories are still available.*

As an extreme example, consider what to do if Yahtzee is the only category left open and after two rolls you have 5 6 6 6 6. No matter what your score is, the best thing to do is reroll the 5. In doing so you will score, on average, $\frac{50}{6}$ *more* points. So instead of keeping track, for each state of the game, what the expected *total* score is, we keep track of what the expected score is for the remaining turns. The average score for the total game would then be the average score for the rest of the game starting at the initial state.

Observation 4.2 *The optimal strategy is sensitive to the* total *score in the upper categories (but not to the distribution of that total among the used categories).*

This is the caveat to the first observation. Because there is a 35 point bonus awarded for totaling at least 63 in those categories, the best strategy does depend on the current score in those categories. For example, suppose the only remaining categories are ones and Yahtzee, we have finished our next to last turn with 5 6 6 6 6 and our upper total is 35. In that case we don't stand to lose much by scoring zero in ones and trying for Yahtzee on the next turn. On the other hand, if our upper total had been 62 then we would want to score zero in Yahtzee because the we have about a 93.5% chance of rolling a one during our next turn and earning our 35 point bonus but only about a 4.6% chance of rolling a 50-point Yahtzee. Note, however, that it is the total score in the upper section that matters and not the distribution of that score (zero ones and four fours is the same as four ones and three fours).

So the information we need in each state consists of: (1) for each category, whether it is used or unused (when playing with Yahtzee Bonuses, Yahtzee is a tri-state category: unused, zero, non-zero); and (2) the score in the upper section. There are 2^{13} possibilities for the former and 64 for the latter (anything greater than 63 is equivalent to 63 since we have already earned our bonus then), so we need $2^{19} = 524{,}288$ states for the beginnings of turns (50% more for Yahtzee Bonuses), better than the original analysis by a factor of about half a million.

These 2^{19} states are the states that represent the *start* of a turn. Each turn also needs states to represent the result of rolls and choices of which dice to keep. The complete graph then consists of 2^{19} widgets, where each widget is constructed as follows:

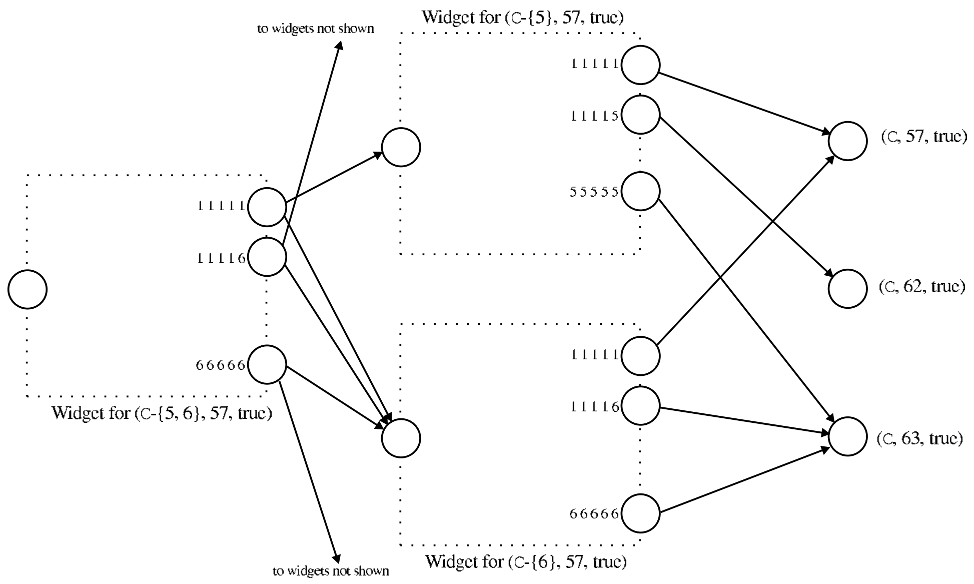

Figure 3: A small portion of the Yahtzee graph showing some of the states in three widgets and some of their interconnections. See Section 5 for an explanation of the notation.

(1) there is a single entry point into the widget that represents the beginning of a turn;

(2) that entry point has edges to states representing the outcomes of the initial roll of the dice;

(3) those states have edges to states representing the choices the player can make of which dice to keep;

(4) edges from group (3) go to states representing the result of the first reroll;

(5) the next group of states is similar to group (3); and

(6) the last group of states is similar to group (2) and represents the outcome of the final roll. States in this group are the exit points of the widget; their outgoing edges go to entry points of other widgets or final states of the game.

There are no edges between different widgets. Note that the outgoing edges from the exit states may differ from widget to widget depending on what categories have been used.

Pattern	Occurrences	Keeps
abcde	$\binom{6}{5} = 6$	32
aabcd	$\binom{6}{1}\binom{5}{3} = 60$	24
aabbc	$\binom{6}{2}\binom{4}{1} = 60$	18
aaabc	$\binom{6}{1}\binom{5}{2} = 60$	16
aaabb	$\binom{6}{1}\binom{5}{1} = 30$	12
aaaab	$\binom{6}{1}\binom{5}{1} = 30$	10
aaaaa	$\binom{6}{1} = 6$	6

Table 1: Patterns of dice and number of ways to choose kept dice.

4.2 Sizing up the graph

To count the number of states and edges in the graph, we need to know the size of each widget. There is clearly one entry point. Counting the number of states in the other groups requires some combinatorics. Each roll corresponds to a bag (or multiset) containing 5 numbers chosen from $\{1,2,3,4,5,6\}$. In general, there are $\binom{n+k-1}{n}$ different size-n bags chosen from k possible elements, so there are $\binom{10}{5} = 252$ possible outcomes when rolling 5 6-sided dice, and so 252 states in groups (2), (4), and (6). There is a state in each both group (3) and (5) for each *partial* roll of the dice – bags of size 5, 4, 3, 2, 1, or 0. There are $\binom{10}{5} + \binom{9}{4} + \binom{8}{3} + \binom{7}{2} + \binom{6}{1} + \binom{5}{0} = 462$ of those. So in total, each widget contains $1 + 3 \cdot 252 + 2 \cdot 462 = 1681$ states.

More important are the number of edges between those states because to figure out the expected value for a given state we need to examine each edge leaving it. It is *not* the case that each state in one group must have an edge to each state in the next group: for example, if you choose to keep 2 3 4 it is impossible to end up with 6 6 6 6 6 after rerolling.

Certainly each state in groups (2) and (4) has no more than $2^5 = 32$ outward edges since there are five dice and for each one you can choose to keep it or reroll it. For many states the number of outward edges is significantly fewer. For example, if you roll 1 1 2 2 2 then rerolling the first die is the same as rerolling the second die, and rerolling the third is the same as rerolling the fourth or fifth. What your decision really is in this case is how many ones to reroll and how many twos to reroll. There are 3 choices of how many ones (zero, one, or both) to reroll and 4 choices of how many twos to reroll, so 12 distinct choices of which dice to reroll. An initial roll of 3 3 5 5 5 would be analyzed the same way – the number of distinct choices of which dice to reroll depends on the pattern of the dice. Table 1 summarizes all the possible patterns (nothing, one pair, two pair, three of a kind, full house, four of a kind, five of a kind), how many times those patterns occur, and how many ways there are to choose which dice to keep for those patterns.

So the total number of edges from the states in group (2) (and (4)) is $6 \cdot 32 + 60 \cdot 24 + 60 \cdot 18 + 60 \cdot 16 + 30 \cdot 12 + 30 \cdot 10 + 6 \cdot 6 = 4368$.

Dice kept	Occurrences	Outcomes
0	$\binom{5}{0} = 1$	$\binom{10}{5} = 252$
1	$\binom{6}{1} = 6$	$\binom{9}{4} = 126$
2	$\binom{7}{2} = 21$	$\binom{8}{3} = 56$
3	$\binom{8}{3} = 56$	$\binom{7}{2} = 21$
4	$\binom{9}{4} = 126$	$\binom{6}{1} = 6$
5	$\binom{10}{5} = 252$	$\binom{5}{0} = 1$

Table 2: Kept dice and number of outcomes of rerolling.

Recall that the states in groups (3) and (5) represent the partial rolls that are left after we decide which dice to reroll. The number of outward edges for these states depends on the number of dice that we have decided to keep. This is summarized in Table 2.

So there are $1 \cdot 252 + 6 \cdot 126 + 21 \cdot 56 + 56 \cdot 21 + 126 \cdot 6 + 252 \cdot 1 = 4368$ edges total from the states in group (3) and the same number of edges from the states in group (5). (Note that this is the same count we got for the edges from group (2) to group (3) since the edges out are essentially a mirror image of the edges in).

Finally, there are 252 edges out of the entry state and at most 13 edges out of each exit state (one for each category that could be chosen). So the total number of edges in each widget is no more than $252 + 4 \cdot 4368 + 13 \cdot 252 = 21000$. There are 2^{19} widgets in the graph, so around 11 billion edges in the graph. This is a number that we can reasonably expect to examine in an hour or so of CPU time, but it would be helpful to reduce the size of the graph even more.

4.3 Unreachable States

We can reduce the number of widgets substantially since some states are unreachable. For example, it is impossible to have an upper total of 4 if the only upper categories used are threes and fives. Figuring out which states are reachable can be solved with dynamic programming.

Let S be a subset of $\{1, 2, 3, 4, 5, 6\}$ (indicating which of the first six categories have been used) and n be a natural number so that $0 \leq n \leq 63$. Define $R(n, S)$ to be true if it is possible to score n points in the upper area using categories in S and false otherwise. Then $R(0, S) = \text{true}$ for all S and $R(n, \emptyset) = \text{false}$ for any $n \geq 1$. For $S \neq \emptyset$ and $n \geq 1$,

$$R(n, S) = \begin{cases} \text{true} & \text{if } (\exists x \in S, k \in N)[k \leq 5 \wedge kx \leq n \wedge R(n - kx, S - \{x\})] \\ \text{false} & \text{otherwise} \end{cases}$$

After computing $R(n, S)$ for all appropriate n and S, we find that 1260 of the 4096 values are false, reducing the number of states and edges by more than a quarter.

5 Computation

We are now ready to describe the computations needed to compute the optimal strategy for solitaire Yahtzee. First, however, we must introduce some notation.

$\mathcal{C}$ is the set of categories, {ones, twos, threes, fours, fives, sixes, three of a kind, four of a kind, full house, small straight, large straight, chance, Yahtzee}. c will stand for elements of $\mathcal{C}$, C for subsets of $\mathcal{C}$.

A Yahtzee state is a triple (C, m, f) where C is the set of categories already marked, m is the total in the upper section, and f is a flag indicating whether the next Yahtzee will earn a Yahtzee Bonus, that is, whether Yahtzee was marked with 50 (the flag is, of course, unnecessary when dealing with the game in which no Yahtzee Bonuses are awarded). Yahtzee states will generally be denoted by the letter S and the set of all possible Yahtzee states will be denoted by $\mathcal{S}$.

$R_{i,j}$ is the set of outcomes when rolling i j-sided dice. The set of rolls we could be left with after choosing which dice to keep is $R_k = R_{0,6} \cup R_{1,6} \cup R_{2,6} \cup R_{3,6} \cup R_{4,6} \cup R_{5,6}$. r will stand for elements of $R_{5,6}$ and r' will stand for elements of R_k ('k' for "keeps"). $\perp$ denotes the empty roll.

$E(S)$ is the expected future score (or "potential") when starting a turn in state S.

$E(S, r, n)$ denotes the potential of state S with roll r and n rerolls remaining. (That is, the potential in the middle of turn begun in state S after one of the rolls of the dice has come up showing r.)

$E(S, r', n)$ denotes the potential of state S having chosen to keep dice r' with n rerolls remaining.

$P(r' \to r)$ denotes the probability that, having kept r', we reroll the remaining dice and end up with r.

$s(S, r, c)$ denotes the score obtained by scoring the roll r in category c, given previous game state S. This score will include any Yahtzee Bonus, but not any Upper Bonus, which we imagine as being awarded at the end of the game instead of the turn in which the upper total first exceeds 62.

$u(S, r, c)$ denotes the additional upper score earned by scoring roll r in category c. Naturally, $u(S, r, c) = s(S, r, c)$ for upper categories and is 0 for lower categories.

$f(S, r, c)$ denotes the new value of the Yahtzee Bonus flag after scoring roll r in category c starting from state S. $f(S, r, c)$ is true if and only if either $S = (C, m, \mathit{true})$ for some C and m, or both $c = \mathit{Yahtzee}$ and $s(S, r, c) = 50$ (that is, we are eligible for the Yahtzee Bonus if we were previously eligible for it or if we have just scored 50 in Yahtzee).

$n(S, r, c)$ denotes the new state after starting in state S and scoring r in category c. $n((C, m, f), r, c) = (C \cup \{c\}, m + u((C, m, f), r, c), f((C, m, f), r, c))$.

The procedure for computing the optimal Yahtzee strategy begins by computing the potential for each final state. If $C = \mathcal{C}$, $E(C, m, f)$ is 35 if $m \geq 63$, 0 otherwise.

For states at the end of turns, the expectation is given by the maximum over all the unused categories c of the score in c plus the potential of the next state:

$$E((C,m,f),r,0) = \max_{c \notin C}(s((C,m,f),r,c) + E(n((C,m,f),r,c)).$$

For states where the player has just chosen which dice to keep, the expectation is given by the weighted average of the expectation of all the states we could be in after the reroll: for $n = 1,2$,

$$E(S,r',n) = \sum_{r \in R_{5,6}} P(r' \to r) \cdot E(S,r,n-1).$$

For states representing the result of the initial roll or the first reroll, the expectation is given by the best expectation of the states we could end up in after choosing which dice to keep: for $n = 1,2$,

$$E(S,r,n) = \max_{r' \subseteq r} E(S,r',n).$$

Finally, for states at the beginning of a turn, the expectation is given by the weighted average of the expectations of all the states we could be in after the initial roll:

$$E(S) = \sum_{r \in R_{5,6}} P(\perp \to r) \cdot E(S,r,2).$$

The order of computation is based on how many categories are filled in a state. By evaluating states with more categories marked before states with fewer categories marked, we ensure that we have all the information needed to compute $E(S)$ when we get to S.

6 Implementation Details and Results

We have coded the algorithm and associated data structures in about two thousand lines of C++ code. Only minor modifications need to be made to allow for minor variations in scoring rules such as Yahtzee bonuses. After about one hour of computation time on an entry-level CPU the maximum possible average score is computed to be approximately 245.87 without Yahtzee bonuses or jokers and 254.59 with.

A naive implementation would attempt to keep track of the expected value from each state in each widget. At about one billion states and eight bytes per expected value, we would need 8GB of storage. While this is not a problem for today's supercomputers, it would surely tax the virtual memory systems of today's desktop computers. Fortunately, we do not need to keep all this information online at once.

Note that in each widget, the computation for states in group 6 depends on other states in group 1 in other widgets. The computations for states in groups 2 through 5 depend only on states in the same widget. So, once a widget is done, we can forget all the information that was computed for it (except for what the ideal strategy is, but that's less that one byte per state in groups 2, 4, and 6, and furthermore only needs to be written and never read). The only information we need to keep around for good is that about the 2^{19} states in group 1, or about 4MB (or 6MB for the game allowing Yahtzee bonuses).

The 4MB table is also good enough to play a game using the optimal strategy. At the beginning of each turn we will need to recompute the optimal strategy for the current widget. We computed the optimal strategy for 2^{19} widgets in an hour, so the strategy for a single widget cam be computed in a fraction of a second. It should be noted that 4MB of data can be stored easily in the main memory of today's desktop computers, but it is still a significant chunk of data for a handheld device. We may still wish to have a handheld device play nearly optimal Yahtzee; in the next section we evaluate some strategies through which we might achieve this.

7 Comparison to Other Strategies

It is possible to use the graph approach to evaluate the performance of any other Yahtzee strategy by working forward from the initial state of the game. For the purposes of this evaluation, we view a strategy as a pair of functions, $f_1 : (\mathcal{S} \times R_{5,6} \times \{2,1\}) \mapsto \mathcal{P}\{1,2,3,4,5\}$, and $f_2 : (\mathcal{S} \times R_{5,6}) \mapsto \mathcal{C}$. f_1 tells us, given a state of the scorecard, a roll of five dice, and a number of rerolls left, which dice to keep. f_2 determines, given a state of the scorecard and a roll, which category to score the roll in. (Note that a human player's strategy may not be representable by functions: faced with the same situation, a human may randomly choose a move based on his or her mood.)

To evaluate the expected score when following a deterministic strategy, we compute, for each state, the probability of reaching that state with a certain score. For this value we write $P(S,n)$ for states at the beginning of turns and $P_{roll}(S,r,n,k)$ or $P_{keep}(S,r,n,k)$ for states in the middle (for the probability of having the scorecard in state S with r showing on the dice, a total score of n and k rerolls remaining). We always start out with zero points at the beginning of the game, so for $S = (\{\}, 0, false)$, $P(S,0) = 1$ and $P(S,n) = 0$ for all $n > 0$.

If we know $P(S,n)$ for a beginning-of-turn state S, we can compute the probability of being in state S and having rolled r in the initial roll:

$$P_{roll}(S,r,n,2) = P(S,n) \cdot P(\perp \to r).$$

The probability of choosing to keep r' after our initial roll is then then the sum over all rolls r such that the strategy tells us to keep r':

$$P_{keep}(S,r',n,2) = \sum_{r \in R_{5,6} \wedge f_1(S,r,2)=r'} P_{roll}(S,r,n,2).$$

Knowing the probability of ending up with each subroll after one choice allows us to compute the probability of facing our second choice with roll r:

$$P_{roll}(S,r,n,1) = \sum_{r' \to r} P_{keep}(S,r',n,2) \cdot P(r' \in R_k).$$

Strategy	Expected Score	Std. Deviation
Yahtzees Only	171.52	68.17
Yahtzees and Straights	202.51	65.90
Greedy	218.05	46.87
Rational Yahtzees	219.86	65.99
Heuristic	240.67	60.90
Better Heuristic	244.87	57.39
Optimal	254.59	59.61

Table 3: Average scores for various strategies.

Repeat for the second and third rolls to obtain $P_{roll}(S, r, n, 0)$, the probability of ending a turn in state S with n points and the dice showing r. We then compute $c = f_2(S, r)$, the category the strategy tells us to score r in, and add $P_{roll}(S, r, n, 0)$ to our running total of $P(n(S, r, c), n + s(S, c, r))$.

After working through the entire graph, the probability of ending the game with score n is given by

$$P(n) = \sum_{f \in \{true, false\}} \left[P((\mathcal{C}, 63, f), n - 35) + \sum_{m=0}^{62} P((\mathcal{C}, m, f), n) \right].$$

Examining this for all values of n yields the expected final score as well as the standard deviation of the final score. The process is computationally expensive: we must compute $P(S, n)$ for each combination of on the order of a million states and 1576 possible scores (the maximum score with Bonus Yahtzees is 1575). The table holding $P(S, n)$ requires several gigabytes and the computation currently takes about 24 hours but nonetheless has been performed for several different strategies, summarized in Table 3.

The strategy described as "Yahtzees Only" is one that might be followed by a player who desires to maximize the total Yahtzee Bonus. It dictates that when choosing which dice to keep, we keep the ones that came up most often. Ties are broken in favor of highest open category, then by highest rank. When choosing which strategy to score a roll in, it dictates a greedy strategy, with ties broken arbitrarily. The next strategy ("Yahtzees and Straights"), is designed with the same goal in mind, but is tempered by the decision to keep a straight if it has been rolled before the turn's end and the category is still open. Even so, the only time this strategy dictates that we chose to keep dice to go for straight instead of Yahtzee is when we already have a small straight and both small straight and large straight are empty. "Rational Yahtzee" is a further refinement of the Yahtzee-minded strategy that will, among other things, keep a doubleton instead of a tripleton if the doubleton's upper category is open and the tripleton's is not.

The rest of the strategies are based on estimates of $E(S)$ that can be computed without using too much storage space (remember, the table of exact values of $E(S)$ occupies several megabytes). The greedy strategy uses $E'(S) = 0$, so that it sees the potential of each state the same, so its goal is always to score the most points on the current turn, with ties broken in favor of the more difficult categories (for example, it will fill four-of-a-kind before three-of-a-kind).

The heuristic strategies attempt to more accurately estimate $E(S)$. The first is based on the expected scores for each category when following the optimal strategy as computed by Tom Verhoeff [2]. $E'(S)$ in this case has three components. The first is the sum of the expected score for each open category. The second is the expected upper bonus, which is computed by assuming that the remaining upper total is normally distributed with variance given by the sum of the variance of the individual categories. Of course this is a gross approximation since the scores in the upper categories are not independent and so their variances do not add, but the performance is impressive nonetheless. The third component is the expected Yahtzee bonus, which is estimated to be 0.

The second heuristic uses larger tables than the previous one but ones that are still much smaller than those needed by the optimal strategy. One table is obtained by computing the optimal strategy for a simpler version of Yahtzee that uses only the upper categories and awards no upper bonus. The other table is obtained the same way but uses only the lower categories. $E'(S)$ is then composed of four components: the expected upper total, which is obtained using the first table, the expected lower total, which is obtained using the second table, and the expected upper and Yahtzee bonuses, which are computed the same was as for the first heuristic.

8 Future Work

Note that this does not maximize the probability of winning a two-player game. For example, suppose your opponent's only open category is ones, he has already earned the bonus, and he is 40 points ahead of you. Suppose your only open categories are Yahtzee and chance and you end your next to last turn with 1 1 1 1 2. When playing to maximize your score you would score zero in Yahtzee and hope for a good roll in chance (the average score for chance when it is the last category left is $23\frac{1}{3}$ while the average for Yahtzee is 2.3014). However, if you score zero in Yahtzee you have no chance of winning the game since you need to score 50 in Yahtzee to overcome the 40 point deficit. If you score 6 in chance you still have a chance to win by getting a Yahtzee in your last turn.

It is not practical to use the method we have for finding the optimal solitaire Yahtzee strategy to find a strategy that will maximize the chance of winning a two-player game, for the amount of information that must be kept is too great. Each state would need to record the state of the player's scoresheet *and* the state of the opponent's scoresheet *and* the number of points currently separating the two. We have already seen that there are about 2^{19} states for each scoresheet. The maximum score without Yahtzee bonuses is 375, so for difference in score we have 751 possibilities (behind by 375 to ahead by 375). Examining reachable states will help us whittle down the numbers (for example, it is impossible for one player to have filled 6 categories while the other player has filled 10, and it is impossible to be 300 points ahead after one turn), but as a gross estimate we see that there are on the order of 2^{48} states to contend with. We currently have no means of dealing with either the time or the space that processing so many states would require.

Things become simpler when you consider a sort of "blind" two-player game. Instead of taking turns playing and being able to monitor the state of your opponent's scoresheet, suppose that you and your opponent play games in separate rooms, and upon completion announce your scores. In such a case you would not be concerned with your opponent's scoresheet or the amount by which you led or trailed since you could not obtain that information. It should be possible to figure out a strategy that will prevail more often than not over someone playing the optimal solitaire strategy. In this case, your current score is important, so the number of states considered will rise dramatically, but the computation should still be within the capabilities of today's machines.

References

[1] Albert H. Morehead and Geoffrey Mott-Smith, editors. *Hoyle's Rules of Games*. Signet, second revised edition, 1983.

[2] Tom Verhoeff. Optimal solitaire yahtzee player: Trivia. http://wwwpa.win.tue.nl/misc/yahtzee/trivia.html, 1999. Visited August 13, 2001.