



YAAAH... TZEEEE!!! BLESS YOU!

A COMPARISON STUDY ON YAHTZEE STRATEGIES

Alessio Van Keulen

ISYE 6644
Simulation

Georgia Institute of Technology
Summer 2023



GO JACKETS!

Contents

Abstract..... 3

Introduction..... 3

Rules 3

Strategies..... 4

 Intermediate Mode..... 4

 Hard Mode 5

The Algorithms 6

 Intermediate Mode..... 6

 Pseudo-Code..... 6

 Steps To Run the Simulation 7

Comparisons and Observations 8

Conclusions..... 9

References..... 10

Abstract

When it comes to beat an AI in the game of Yahtzee, it very much depends on your strategy, luck, and what algorithm you are playing against. Even if you are a very lucky player, and you deploy your best strategy, a statistics algorithm should take into consideration all the possible scoring possibilities to guarantee the most optimal outcome and outperform your best total score. This paper will test two different algorithms, representing two different levels of AI difficulty: “intermediate” and “hard”. The results will be compared and charted.

Introduction

Yahtzee is a popular dice game where a player can score points based on the outcome of a roll of five dice. Although seasoned players may have perfected their strategy to maximize their win, the absolute best strategy to always guarantee the upper hand over several matches is not so obvious. The task of developing the optimal Yahtzee strategy has already been tackled by James Glenn in 2006, when he produced an algorithm that could individuate the statistical best choice for every state of the game and deliver an average max score of 254.59. Using this score as a basis for the “hard” AI difficulty, this project will propose an alternative intermediate strategy, compute its average max score, and finally compare results.

With Yahtzee being played all over the planet, slightly different variations, and rules of the game lead to different strategies. The algorithms discussed in this paper adopt the official version of game and its rule set.

Rules

- The goal of the game is to score points based on the outcomes of a toss of 5 (fair) dice.
- Each toss may result in a combination of numbers that unlocks a scoring objective marked on a scoring card provided to each player.
- Once a scoring objective is used, the player must choose another of the remaining scoring objectives.
- Each turn, a player is allowed to roll the dice a maximum of 3 times, so they have a better chance at achieving the desired scoring objective.
- After every roll, the player is allowed to “lock” one or more dice and re-roll the rest (much like a reel on a slot machine).
- The player with the highest total score is the winner.
- The scorecard is divided into two sections. A bonus of 35 points is added to the upper section’s total, if the total is at least 63 points.
- Every additional Yahtzee is considered a bonus adding 100 points to the lower section’s total.














UPPER SECTION	HOW TO SCORE	GAME #1
ACE  = 1	COUNT AND ADD ONLY ACES	
TWOS  = 2	COUNT AND ADD ONLY TWOS	
THREES  = 3	COUNT AND ADD ONLY THREES	
FOURS  = 4	COUNT AND ADD ONLY FOURS	
FIVES  = 5	COUNT AND ADD ONLY FIVES	
SIXES  = 6	COUNT AND ADD ONLY SIXES	
TOTAL SCORE		
BONUS <small>IF TOTAL SCORE IS 63 OR OVER</small>	SCORE 35	
TOTAL <small>OF UPPER SECTION</small>		
LOWER SECTION		
3 OF A KIND	ADD TOTAL OF ALL DICE	
4 OF A KIND	ADD TOTAL OF ALL DICE	
FULL HOUSE	SCORE 25	
SM. STRAIGHT <small>(SEQUENCE OF 4)</small>	SCORE 30	
LG. STRAIGHT <small>(SEQUENCE OF 5)</small>	SCORE 40	
YAHTZEE <small>5 OF A KIND</small>	SCORE 50	
CHANCE	SCORE TOTAL OF ALL 5 DICE	
YAHTZEE BONUS	 FOR EACH BONUS  SCORE 100 PER	
TOTAL <small>OF LOWER SECTION</small>		
TOTAL <small>OF UPPER SECTION</small>		
GRAND TOTAL		

Figure 1. Yahtzee Scorecard

Strategies

In this study two different strategies will be tested over 5,000 games. The results will be charted and used to compare the algorithms, and rank them on a scale of difficulty, if a human player were to play against them.

Intermediate Mode

In “Intermediate Mode” the algorithm attempts at achieving the highest grand total score implementing a conservative strategy that involves two important ideas:

1. Minimizing the scoring error at the end of any given turn.
2. De-prioritizing low scoring objectives.

A scoring error is considered any scoring that doesn't contribute to the 63 minimum points necessary in the upper section to unlock the 35 extra bonus points. Playing against other human players, it has become evident how the upper section bonus constitutes an advantage point, and often decides the results of the game. Consider this example:

TURN #	1 ST DICE TOSS	2 ND DICE TOSS	3 RD DICE TOSS	SCORED
1	1,1,2,3,6 Lock the Sixes	1,1,2,6,6 Lock the Aces and Sixes	1,1,6,6,6	Scores Sixes 18

Even though in turn #1, the 1st roll yielded two Aces, the algorithm choses to lock the Sixes because they imply a higher potential gain: 3 Sixes are a better result than 3 Aces, as they contribute better to the points necessary in the upper section to unlock the bonus. At the 3rd roll, the algorithm scores 18 in the Sixes. Alternatively, it could have chosen to score Full House (25 PTS), but in this strategy Full House is considered of less priority than scoring 18 in the Sixes because it does not contribute to the upper bonus, and it can be achieved with other combination of numbers.

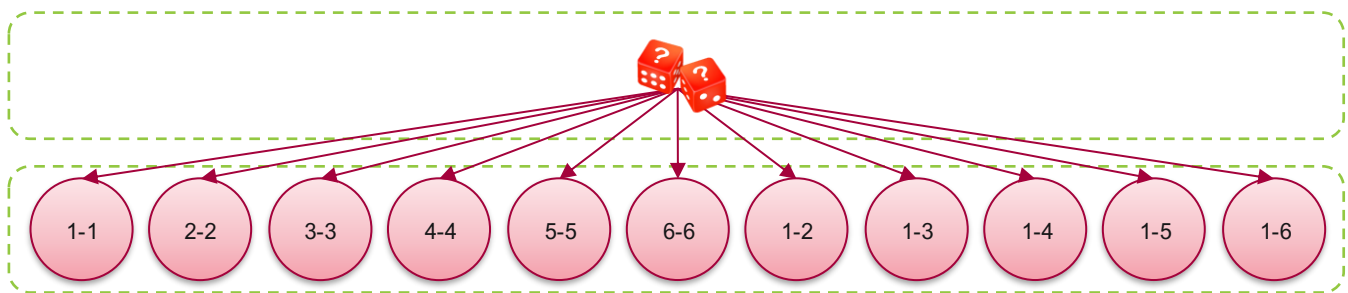
De-prioritizing low scoring objectives involves trying to avoid scoring Aces and Twos at early turns in the game, as the max potential score for those objectives is 4 PTS (for Aces), and 8 PTS (for Twos). Note that it is not 5 PTS and 10 PTS respectively, because that would be a Yahtzee (50 PTS) instead. Excluding Aces and Twos from early scoring attempts allows for a larger margin of error in case the player fails to collect higher yielding objectives, e.g., Fives and Sixes. In the following example, the player is about to roll their 8th turn. At this point they have already scored Twos (8 PTS), Threes (9 PTS), Fours (12 PTS), and Sixes (18 PTS) in the upper section, but they still need Aces and Fives. The current upper section total is 47 PTS. The current scoring objective is Fives, as rolling at least 3 fives would grant 15 PTS, bumping up the upper section's total to 63. That is the minimum score needed to unlock to upper section's bonus: a good choice!

TURN #	1 ST DICE TOSS	2 ND DICE TOSS	3 RD DICE TOSS	SCORED
8	1,1,4,5,6 Lock the Fives	1,1,5,5,6 Lock the Fives	1,1,3,5,5	Scores Aces 2 PTS

Because the Aces were de-prioritized earlier in the game (see above example: turn #1), they are still available. The player attempted at collecting 15 PTS with the Fives, but only totaled 10 PTS (2 Fives). If they were to choose to score the Fives, they would compromise the upper bonus; player should choose to score the Aces instead.

Hard Mode

In “Hard Mode” the algorithm computes the statistically best option for each single state of the game. The state of the game is any single dice toss at which point the player is tasked with choosing either to roll the dice again, or to score some points. Each decision leads to the next state in the game. Consider the network-graph below representing the outcomes of a toss of two dice:



- This is a simplified game that allows for only one roll.
- There are only two states (dashed boxes): an initial state in which dices can be rolled, and the final state listing all the possible outcomes for the roll.
- Each outcome has an associated probability. See the below examples:

COMBINATION	PROBABILITY	EXPLANATIONS
1 - 2	~ 0.06	$\frac{1}{6} \times \frac{1}{6} \times 2$
6 - 6	~ 0.03	$\frac{1}{6} \times \frac{1}{6}$

- If the game were to allow for another roll, a third dashed box (third state) would appear, displaying all the outcomes associated with the second roll.

In the real game of Yahtzee, the associated network-graph would be much larger, involving many states. In fact, Glenn (2006) calculates over 209 billion different states of the game, with the final state listing all possible outcomes derived by random chance or by player choices. The algorithm carefully computes the projected value for each state and acts based on which choice yields the greatest average total score.

The Algorithms

In this paper, we won't discuss the details of the algorithm that is behind the creation of the Hard Mode. For more information about this, you can read the white paper, "An Optimal Strategy for Yahtzee" by James Glenn (2006).

Intermediate Mode

For this project, Intermediate Mode has been developed using JavaScript programming language. The reason for this choice is the almost ubiquitous availability of the language itself, as being built into almost any internet browser, and its great speed. A simulation run of 5000 games was completed in ~ 2157ms, which is a good performance given that the algorithm may have not been written in the most efficient of ways.

Pseudo-Code

In this algorithm each possible scoring objective is assigned a priority. The priority is focused around collecting the upper bonus of 35 points. These are the rules:

- Higher priority goes first.
- When priority is the same, algorithm locks the best non-scored numbers and tries to maximize those, alternatively it tries to score an objective in the lower section of the game.
- The priorities change depending on what has been scored previously.
- Each scoring objective is represented by an array whose index is mapped to the related scoring objective (index = 0 is not used):

```
var completed_goals = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
var scores = [null, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
```

- At the same time, the amount of points for each scoring objective is stored in another array of the same size:

The code is split into several functions, each of which addresses a particular action needed in the game. Here are a few of the most important ones:

<i>Function name</i>	<i>Has Parameters</i>	<i>Task</i>
Check_rolls()	new_roll, previous_roll	At each dice roll will check for what makes sense to hold in order to achieve a specific scoring objective
Score()	final_combination	Takes the dice from the last roll, and takes care of scoring the best possible score
Consecutive_dice()	combination	Takes the current combination of dice at any given roll, and returns all the consecutive dice if any.
Score_mistake()	Tgt, times	If a particular scoring objective is missed, this function takes care of scoring the lowest possible mistake. E.g. the player was trying to achieve large straight with 1,2,3,4,6. This function will score a 1 in the Aces.


```
// Should check the outcome of a roll, and hold dice if necessary
const check_rolls = (new_roll, previews_roll) => {
  var counters = [0, 0, 0, 0, 0, 0];

  var current_combination = previews_roll.concat(new_roll);
  current_combination.map((x) => (counters[x - 1] += 1));

  var set = new Set(current_combination.sort());
}
```

Figure 1 – check_rolls() function

The check_rolls() function takes the currently rolled dice and the previously held dice to make the new current combination. From here a set is created to which dice are unique, and all the repeating dice are counted in a local array variable “counters”.

Steps To Run the Simulation

1. Find Yahtzee_intermediate.html file available in this project directory, specifically under \game\yahtzee_intermediate.html.
2. Run the file in your browser. Make sure to open the developer console: anywhere on the page Right-Click > Inspect > Console. This will provide you with a debugging area which will prove helpful in case you’re interested in debugging the outcome of certain actions.
3. The game’s interface will allow you to play one or more games:

Simulate	Console Output	Comments
One dice roll	Yes	
One full game	Yes	
6 full games	Yes	Console will lag some time
5000 full games	No	Will prompt a CSV download link

4. In this study, we will run our Yahtzee simulation 5000 times, and visualize the results.
5. Optional, it is possible to run more simulations, by changing the parameter of the function that generates the simulation:

```
538 | <button class="button-30" onclick="run_simulation(5000)">Simulate 5000 Games</button>
```

NOTE: Upon running the large simulation, a prompt to download the data will appear. This data was then imported into excel, along with the data generated by the Hard Mode algorithm, and then used to draw the comparison charts.

6. Optional: You can also view the Hard Mode algorithm by running the file ComputerYacht.exe under \game\ ComputerYacht.exe (courtesy of The Yahtzee Manifesto)

Comparisons and Observations

After 5000 runs, it emerges that the Hard Mode, powered by the Markov-Chain algorithm, seems to have the upper hand in comparison with the Intermediate Mode.

General Statistics	Hard Mode	Intermediate M.
Number of Games Played	5000	5000
Total Score	1344399	1041531
Yahtzee %	46%	28%
Upper Bonus %	76%	42%
Min Score	85	105
Max Score	737	674
Avg Score	269	208
Variance	5385	3059
Std Dev	73	55

Figure 2a – General Statistics

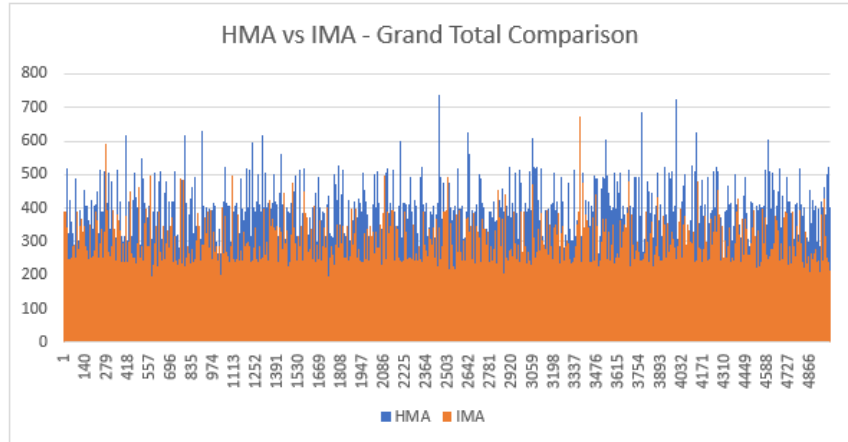


Figure 2b – Hard vs Intermediate Mode Algorithm Grand Total

Although the Hard Mode outperforms the Intermediate Mode in almost all statistics (Fig. 2a), it is interesting to notice how the latter has a higher minimum total score, and a smaller variance. This falls along with the expectation, as the first algorithm follows a mathematical approach with the goal of maximizing the grand total, whereas the second has the goal making the best scoring choices in order to achieve the upper section's bonus and minimizing the error. In Figure 2b, we can see the grand total for both algorithms compared together. From the Box-Whiskers plots (Fig. 3) we can see where the concentration of points is.

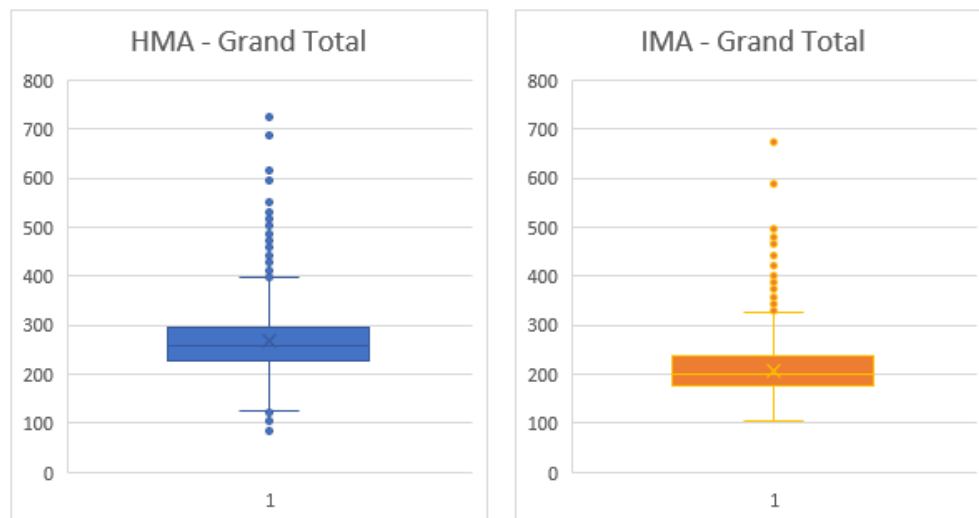


Figure 3 – More insights about grand total

For the Hard Mode algorithm, the results tend to be slightly higher than the Intermediate Mode: centered around [220,300] total points vs [180,240]. However, the Hard Mode seems to display some very low scoring games, identified as outliers, that the Intermediate algorithm never seem to reach.

Conclusions

As expected, the study showed that the Hard Mode is the strongest algorithm capable of solving a Yahtzee match and scoring great results. The Intermediate Mode reflects the strategy of a seasoned player; yet the choices that would seem more logical for a skilled player could be interpreted as fallacies in the calculation of the associated probabilities of outcome, and therefore discarded by the mathematical algorithm. As being the author of the Intermediate Mode algorithm, I know that some of the decisions made into which dice are best to hold, when is it appropriate to score a mistake, and by how many points can be tweaked to achieve better results. However, these results will still not guarantee the upper hand against the hard mode, as the simulation of many games have demonstrated.

References

Glenn, J. (2006). An Optimal Strategy for Yahtzee. Retrieved from:

http://gunpowder.cs.loyola.edu/~jglenn/research/optimal_yahtzee.pdf

Instructions 1 or More Players. Retrieved from: hasbro.com/common/instruct/Yahtzee.pdf

The Yahtzee Manifesto. Yahtzee Odds and Probability. Retrieved from:

<https://www.yahtzeemanifesto.com/yahtzee-odds.php>