KTH Computer Science
and Communication

OPTIMAL YAHTZEE

Nils Dahlbom Norgren

nilsdn@kth.se

Kungshamra 83 LGH 1007, 17070 SOLNA

0707606440

Philip Svensson

phisve@kth.se

Vikingagatan 7A, 11342 STOCKHOLM

0708928322

Degree Project in Computer Science, First Level

KTH Royal Institute of Technology. CSC

Supervisor: Vahid Mosavat

Abstract

The main purpose of this project is to create an algorithm that optimizes and plays Yahtzee with a high average score. The algorithm uses probability theories and rule based heuristics to achieve this. The Yahtzee game has been analysed from a probability point of view, the goal is to optimise every turn. The algorithm and how it has been used is presented in the paper as well with the results and the differences with other strategies. The result in this project was very successful with an average score of 211,04 on 100000 runs.

Sammanfattning

Syftet med detta projekt är att skapa en algoritm som optimerar och spelar Yahtzee med ett så högt medelvärde som möjligt. Algoritmen baseras på sannolikhet och regel-heuristik. Algoritmen som har använts, och resultat av denna implementering, presenteras i denna kandidatuppsats tillsammans med jämförelser med andra kända sätt att optimera Yahtzee. Resultatet av implementationen gav ett medelvärde på 211,04 poäng över 100000 körningar.

Statement of Collaboration

During the development of the program the two group members worked alongside each other, switching places on the same computer. The report was written together but each group member had responsibility over certain parts of the document.

Group member	**Chapter**
Nils Dahlbom Norgren	Abstract, Approach, Tables & Graphs, Conclusion.
Philip Svensson	Introduction, Background, Results, Discussion.

Chapters omitted have been written in collaboration between both group members.

Table of Contents

Introduction **6**
Background and the basics of Yahtzee **6**
Rules of Yahtzee **6**
Problem Statement **7**
Limitations **7**
Background **8**
Existing research **8**
Tom Verhoeff - How to Maximize Your Score in Solitaire Yahtzee 8
James R. Glenn - Computer Strategies for Solitaire Yahtzee 8
Approach **10**
Computer Algorithms **10**
How many dice has to be rerolled? **10**
Calculate the possibility of achieving each category **11**
Combining the two above and get the weighted value **13**
Choosing the category/dice that gives the highest expected score **13**
Results **14**
Discussion **16**
Comparison with the optimal algorithm **17**
Conclusions **18**
References **19**
Appendices **20**
Appendix A. The Yahtzee Scoreboard **20**
Appendix B. A Detailed Comparison between this work's and Tom Verhoeff's algorithm. **21**

Introduction

Background and the basics of Yahtzee

Yahtzee is one of the oldest games that exist. It is remarkable how a simple dice game can be played by so many generations of people all around the world. Everyone that can roll a dice and count can play the game. Like every other board game you can play just for the fun of it, but you can go one step further and try to calculate the probability of getting that certain dice roll. When you calculate the probability of getting a certain category it is pretty simple and straightforward. Although, if the player takes the probable final score into consideration when calculating; the game can be played even better. If taken into account which categories are already taken the calculations are starting to get more and more complex. While we can do simple probability calculations in our brain we cannot hold a complex equation in our mind without putting it on paper and even then it will take a lot of time. This is where it gets interesting to include a computer to calculate probabilities. The game will be taken to a whole new level and one no longer depends only on luck and intuition.

Playing solitaire Yahtzee is when someone plays against himself to achieve the highest score possible, the outcome of the game does not depend on any other players. In every roll in every game calculations or guesses will have to be made to get the highest score at the end of the game.

Rules of Yahtzee

The player starts out with a blank scoreboard and five dice [1]. The goal is to fill the different categories on the scoreboard according to the rules. There are three rolls per round and after that, the player have to set out points on one of the unused slots in the scoreboard. There are 13 rounds played per game. If the dice does not fit into any unused category, "zeroing"[1] one of the unused slots will be obligatory. This means that you will not get any points from that category at the end of the game. The slot that gives the most points is named after the game itself, Yahtzee, which gives 50 points. The reader can see the scoreboard with rules in Appendix A.

---

[1] With "zeroing" means that a category will be taken out of account for the rest of the game and the score in this category will be set to zero.

The maximum score in a Yahtzee game is 375. This means getting all categories filled with the maximum amount. This score is based on getting (in the Upper Section) five aces (5 points), five twos (10), five threes (15), five fours (20), five fives (25), five sixes (30), (in the Lower Section) five sixes (Three of a Kind, 30 points), five sixes (Four of a Kind, 30 points), three of a kind and two of a kind at the same time (Full House, 25 points), four dice with sequential faces (small straight, 30 points), five dice with sequential faces (large straight, 40 points), five of a kind (Yahtzee, 50 points), five sixes (chance, 30 points) and finally a bonus of 35 points. The bonus is dealt to the player who gets a total score of 63 or above in the upper section of the scoreboard meaning aces, twos, threes, fours, fives and sixes.

Problem Statement

Within this work the goal is to test if a computer can choose the best category to play on each turn and therefore achieve an average score that is as high as possible. The goal is to make a program choose a category through probability-based methods and also some set of rules, i.e. a heuristic. With probability based methods means to make the computer program run calculations on the last thrown set of dice and calculate what is the probability of getting another certain set of dice.

Due to limited time the goal will be to use turned based probabilities rather than the probability of the whole run. The results will be compared and we will see how big of a difference there is. This means that the computer will have to make due with information regarding what the faces of the dice is and which categories are available. If possible a comparison between human playing and the results of this work will be done and analysed.

Limitations

The limitation of this project is mainly time. There will only be time to try and get one efficient way of getting a result that is as good as possible. Another limitation is that the program should be able to run on a standard desktop computer and not taking too much time.

Background

Existing research

The idea of optimising the Yahtzee game with the help of mathematics and computers has been tried several times before. There are a lot of researches on this subject; most of them discuss optimising a solitaire Yahtzee game when the goal is to achieve the highest average score possible. The problem of optimising Yahtzee has been solved, which means that it has been a maximum average score that has been proven to be correct and the highest achievable.

Tom Verhoeff - How to Maximize Your Score in Solitaire Yahtzee

Tom Verhoeff research [2] is often said to be the optimum strategy for playing solitaire Yahtzee in any game. Tom uses a method, which he is calling "Bag Model". This model is quite complex and there is no need to explain all of it in this paper. The interesting thing about this method is that he builds a graph that consists of all the possible outcomes of a game. When rolling the first roll of dice the computer will check in the graph for the highest possible final outcome. This is why this method is superior to all other methods that either consists of rules and heuristics or simple probabilities. The rules used in Tom Verhoeff's work differs a little bit from the work in this paper. The official rules of Yahtzee is not set in stone and therefore you may choose if you want to play with Yahtzee-bonus or not, Tom Verhoeff chooses to do so.

James R. Glenn - Computer Strategies for Solitaire Yahtzee

This paper [3][4] discusses the different strategies that have been used in work by others. In their tables the strategy for optimal score is 254.59 (mentioned above in the text about Tom Verhoeff's method) and some other strategies with lower results are shown as well. All of these methods use a large graph with all the possible outcomes and checks that graph for information on which category to choose. The graph in the optimal strategy consists of half a billion positions. In the optimal strategy the graph becomes this large because it differs from the other strategies in one crucial point. The optimal strategy calculates what the probable final score will be instead of just calculating what the highest score is that you can get on that certain turn.

There are some discussions about strategies that can be used by humans. These are mostly examples on where humans choose the wrong dice to save but also how you can train your Yahtzee skills. For example the author has done some research on how humans choose which dice to keep. Humans often strive for getting Yahtzee; we give up a lot of points to be able to get Yahtzee. The chance of getting Yahtzee in three rolls is about one in 22, you can easily see that the chances are not that good when considering that there are only 13 attempts per game. The other big mistake people make is zeroing the hardest-to-get categories (except Yahtzee) when this may not be the smartest thing to do. The training on the other hand is quite simple, James R. Glenn is suggesting a method where a human plays a number of Yahtzee games and then compares the average score with the average score on the optimal Yahtzee. If you do this, the author suggests that you will know which categories that you overvalue and which you undervalue.

It is noteworthy that James R. Glenn has in his own work computed an optimal algorithm to play Yahtzee as well. The optimal score was the same as the one Tom Verhoeff computed but Glenn made a calculation for the rules used in this work and the optimal score was 245.87.

Approach

The first problem is that the computer has to make the right choice when placing the score in the right category, e.g. if there are three fours, do you fill the fours in the top scoreboard or do you fill three of a kind. Although, a bigger and more complex problem is to get the computer to chose which dice to keep from the first and second roll. In the last described problem it is not that hard to make the computer save the dice to be able to fill the categories that only consists of only one face.

One of the most important things that exist in optimizing Yahtzee is to calculate the probability [2]. A Yahtzee dice consists of as known six faces with the same amount of possibility to show up on each dice on each roll. After the first roll we have an outcome existing of five faces, a subset of the dice is chosen to be re-rolled. After the first roll you have an outcome existing of five faces. The approach in this stage of the game is to calculate the percentage of getting a chosen combination in the next roll, or the two next rolls.

Computer Algorithms

The basic idea of the solution that has been used consist of four larger parts, we have:

- If there is one or two throws left, calculate how many dice that has to be rerolled to get the each combination
- Before each throw calculate the percentage of getting each category combination in the following one or two throws.
- Combining the percentage calculation with the value one can get if scoring that combination.
- When the last throw has been made, put the obtained value in the category that gives the highest score.

How many dice has to be rerolled?

After the first and second roll on each turn the algorithm calculates the shortest distance to achieving every one of the remaining categories. That means that it calculates the highest amount of dice that can be rerolled to achieve all categories remaining. For example, on the first roll a player rolls the dice 1,3,4,6,6. The amount of dice that has to be rerolled for achieving the shortest distance to certain categories is for example:

- Three-of-a-kind: Reroll 1,3 and 4 to get another 6.
- Full house: Reroll 1 and 3 to get either 4 and 6 or 4 and 4.
- Small Straight: Reroll 6 and 6 to get a 2
- Large Straight: Reroll 6 and 6 to get a 2 and 5 or reroll 6 and 1 to get 2 and 5

When dealing with a Full House, things are a little bit complicated. A full house consists of a two-of-a-kind and a three-of-a-kind. The complicated part is not which dice to save because of course the two faces of which there are the most of will be saved. E.g. if there is a series of dice which is 1,1,3,4,5, the ones and also one of the threes, fours or fives will have to be saved to achieve the shortest distance. It does not matter if the face shows a higher number; the category is 25 points nevertheless. The most complex part comes next, if you save the ones and the five you have two possible outcomes that fit into the Full House category, either you will get one more one and one five, or you can get two fives and one one. This makes the calculations on this given category more complex than the ones where you want only one certain face.

All the things mentioned above will be used when the probability calculations will take place.

Calculate the possibility of achieving each category

The main calculations done in this program is to get the probability of each possible outcome after the next roll. There are two calculations like that done during one turn. The first is to calculate the possibility of scoring in a category in the two following rolls; the second is for the next roll. There are two different mathematical equations for calculating these; there is also one for calculating the binomial coefficient.

Equation 1. Binomial coefficient, total amount of ways for picking i things out of j. [5]

$$C_{i,j} = \frac{i!}{(i-j)!}$$

Equation 2. Probability of going from i to j dice of the same face. [5]

$$p_{i,j} = C_{i,j} * p^{i} * (1 - p)^{i-j}$$

If Equation 2 is used for all combinations of i and j where i and j is bigger then zero and smaller then six, the matrix in table 1 is received. It shows the probability of going from i to j dice of the same face.

From / To	1	2	3	4	5
1	0.4823	0.3858	0.1157	0.0154	0.0008
2	0	0.5787	0.3472	0.0694	0.0046
3	0	0	0.6944	0.2778	0.2778
4	0	0	0	0.8333	0.1667
5	0	0	0	0	1

Table 1. Probability transition matrix for one roll, for going from i to j dice with the same face.

Equation 3. Probability of going from i to j dice of the same face in two rolls. [5]

$$p^{2}_{i,j} = \sum_{k=1}^{5} p(i,k) * p(k,j)$$

If Equation 3 is used for all combinations of i and j, the matrix in table 2 is received. It shows the probability of going from i to j dice of the same face in two rolls.

From / To	1	2	3	4	5
1	0.2326	0.4093	0.2702	0.0792	0.0088
2	0	0.3349	0.4421	0.1945	0.0285
3	0	0	0.4823	0.4244	0.0934
4	0	0	0	0.6944	0.3046
5	0	0	0	0	1

Table 2. Probability transition matrix for two rolls, for going from i to j dice with the same face.

Combining the two above and get the weighted value

The last step before choosing the dice is to combine the probability of getting a certain category with the expected score that it will give. The outcome is a weighted value, which is used for choosing between the different categories

Choosing the category/dice that gives the highest expected score

Last in the algorithm is choosing what category to put the result in or which dice to be rerolled. The complex part is when combining several faces to fill a category. These categories are small straight, large straight and full house. When it comes to the Small Straight you want four sequential faces. Say you have two pairs of two sequential faces, e.g. 1,2 and 4,5, in this case it would be better to save the 4,5 pair because you can get both 2,3 and 3,6 instead of just 3,4 in the first pair. Although, when calculating for which dice to save and which to throw again the preferable state is to be as few dice from the goal as possible. With that last thought the saved dice would be the 2,4 and 5 and then there are two dice left which only one have to be a 3. See picture 1. With the large straight there is a very similar problem on which dice to save but it is a little bit easier to solve because all the dice have to show different faces and there can only be a six or a one, not both.

If there are one or more throws left on the turn the algorithm chooses to keep the dice of the category, which has the highest weighted value. If all three rolls have been made the algorithm chooses to put the result in the category that gives the highest value.

Results

During the project there were a number of runs that were aimed to achieve the highest possible average score over a large number of runs. The final and best one gave an average score of 211.04. Running the program for 100000 times and taking the average score of all runs achieved this score. A similar test was made but with only the probability calculations; e.i without any set of rules. That run gave a final average score of 204.75 in 100000 runs. During the phase of fine tuning the program 10 000 runs were made and they showed very little difference between each other and therefore a number ten times as high as that was used for the final test.

Category	Average Score
Ones	1.35
Twos	4.28
Threes	6.83
Fours	9.91
Fives	12.99
Sixes	16.26
Bonus	5.25
Three-of-a-kind	21.05
Four-of-a-kind	10.41
Full House	20.64
Small Straight	29.91
Large Straight	35.21
Yahtzee	17.95
Chance	19.00
Grand Total	211.04

Figure 1. Average score for 100000 runs.

The average score of every single category was also calculated as can be seen in Figure 1. This is what an average run would look like, if that existed. There are a few categories that stands out, for example the Bonus is just five in average, which means that it is only achieved to get the upper sum over 63 in about one in seven tries. You can compare that probability with the ones in Small and Large Straight, which is worth about the same score, happens almost every game. In the Discussion you can see a comparison between these results and the ones of Tom Verhoeff's method.

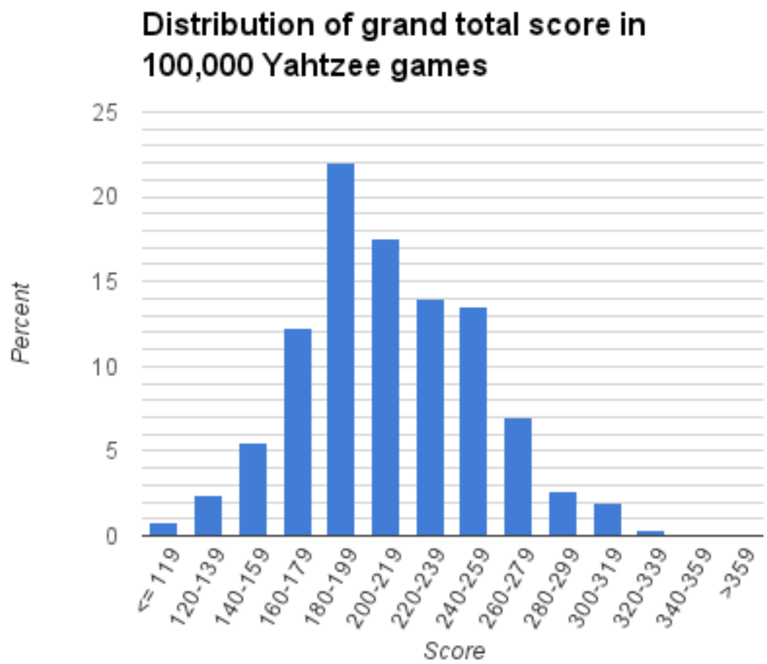

Figure 2. Distribution of grand total score in 100000 Yahtzee games.

The distributions of the final scores are well spread through the whole span as can be seen in Figure 2. The highest score during these runs was 344 and the lowest 62. As can be expected when working with random data and probabilities we have pretty much a “normal distribution”. Noteworthy is that almost 60% of the final scores was over 200 points.

Discussion

As the reader may have noticed, the results in this work are not as good as the optimal version of Yahtzee. Although the goal of this work was to see if it was possible to construct a program that plays Yahtzee as good as possible. The focus laid on using quite non-complex methods and use the computer to be able to handle large amounts of data and to calculate probabilities very fast. The results did show that it is possible to do that. One of the goals was that the program should beat a human playing. No sufficient data was found on average scores for humans, which is very unfortunate.

There are a few reasons that this program did not reach the optimal score. The first one is that the program only calculates the best possible outcome of that specific turn whilst the optimal way of doing it is to calculate the best possible outcome of the whole game. An example of this is that the program will often try to score on Small Straight because this is relatively easy to get. If the first set of dice is 1,3,4,6,6 there is only one dice from getting a Small Straight (30 points) and also only one dice from getting Three-of-a-Kind which will give approximately 25 points. The program does also calculate the probability of getting the sixes in the Upper Section, which we already got two of but 12 points is not a lot to put in a category when you can get 30 points. Perhaps it would be better to save the sixes and try to get a good start on the way to getting the Upper Section bonus. This is pretty hard to handle, of course rules can be made so that the computer will prioritise the Upper Section. A try to do this was done but with little success. The problem is that there is not a perfect strategy for playing Yahtzee except for calculating all the possible outcomes as is done in the optimal play. This is the main reason of the score difference between this average score and the optimal one. The reader can see this in a comparison further down in this section (Figure 3). Here you can see that in Tom Verhoeff's work, he gets an average score of almost 24 points on the Upper Section Bonus whilst the average score is just over 5 points in this work.

Another reason for the relatively low score is that the Chance category is not handled at all. Even in the strategy for humans that James R. Glenn recommended [3] you should not put a score of lower than 22 in the Chance category. This has been tested in this program but with worse results than without a rule for the Chance. The problem with rules is that they do something to make a certain category better and then some other category will be affected.

Yahtzee is a rational game, meaning that you can through calculations be sure to make the best possible decision in each turn. As Tom Verhoeff and others have shown there is an optimal way of playing Yahtzee. But because of this the solution of this work lacks a bit in rationality. Although the computer will choose what seems to be the best solution at hand. The final average score shows that in some of those cases, the computer did not choose the optimal play.

In this work the throw of a dice was calculated with the random generator that is built in in Java. This random generator is quite good but you will never get this "as random" as in real life. This is nothing to dwell over because the work that is compared with this work is also using a computer to randomise the throw of a dice.

Comparison with the optimal algorithm

The differences between Tom Verhoeff's optimal game [2] and this work's algorithm are easily seen in Figure 3. There are two categories where this work's algorithm scores an average higher than Tom Verhoeff's algorithm; Small and Large Straight. There is also one where the optimal algorithm scores distinctly higher than this work's algorithm; the Upper Section Bonus. The average difference between the two algorithms is 2.48 points per category (final score not included). If the Upper Section Bonus is taken away from these calculations the average score differs with just 1.22 points per category. The difference on the average score on just the lower section; e.i from Three-of-a-kind to Chance is 0.67. The upper section average difference is 1.87.

To see the exact difference in each category, see Appendix B.

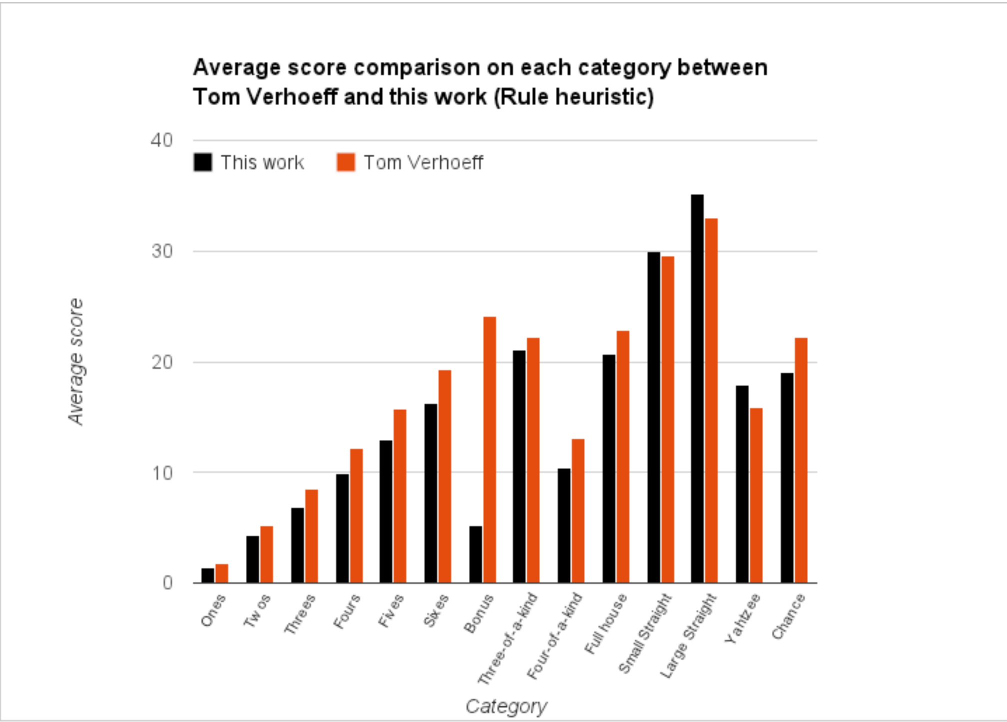

Figure 3. Average score comparision on each category between Tom Verhoeff and this work.

Conclusions

In the result part of this paper it was evident that the result got much higher when a few simple rules were added into the equation. The interesting part is that a heuristic approach together with rules can most likely get a lot higher results if some more strategies were tested. The focus of this work was on rational thinking and behaviours; it was proven that these methods work with Yahtzee. This means that Yahtzee is a rational game but you will never know if you make the rational choice every time.

As stated an optimal algorithm for solving Yahtzee was not constructed but through research an optimal algorithm was found and therefore verifying that there is an optimal way of playing Yahtzee. A comparison between the optimal results and the results of this work was done and the differences are very distinct. The big difference is the final score, which differs about 30 points in average.

One of the goals was that the computer should beat a human player. No sufficient data was found on this matter and therefore there is no way to be absolutely sure that the computer will actually be better than a human player in every game.

The next step in the process of achieving an optimal algorithm for playing Yahtzee would be to either try or make something similar to Tom Verhoeff's solution with the use of large graphs. The other way to improve the final average score would be to improve the algorithm of this work. The first thing to change would be to improve the scoring on the Upper Section of the Yahtzee table and therefore getting the Bonus more often and thus reaching a higher average score.

References

[1] Rules of Yahtzee. Website. URL http://www.hasbro.com/common/instruct/Yahtzee.pdf

[2] Tom Verhoeff. How to maximize your score in solitaire Yahtzee. Eindhoven: Technische Universiteit; 1999. URL http://www-set.win.tue.nl/~wstomv/misc/yahtzee/yahtzee-report-unfinished.pdf

[3] James R. Glenn. Computer strategies for solitaire Yahtzee. Maryland: Department of Computer Science Loyola College in Maryland; 2007. URL http://ieeexplore.ieee.org.focus.lib.kth.se/stamp/stamp.jsp?tp=&arnumber=4219034

[4] James Glenn. An optimal strategy for Yahtzee. Maryland: Department of Computer Science Loyola College in Maryland; 2006. URL www.cs.loyola.edu%2F~jglenn%2Fresearch%2Foptimal_yahtzee.pdf&h=SAQGdLxx4

[5] Gunnar Blom, m.fl.: Sannolikhetsteori och statistikteori med tillämpningar. Studentlitteratur, Lund, 2005 (5:e upplagan).

Appendices

Appendix A. The Yahtzee Scoreboard

Category	Rule	Score
Aces	N/A	The sum of all aces
Twos	N/A	The sum of all twos
Threes	N/A	The sum of all threes
Fours	N/A	The sum of all fours
Fives	N/A	The sum of all fives
Sixes	N/A	The sum of all sixes
Sum of upper section	The sum of all scores in the categories from “Ones to Sixes”	Sum of all the above scores
Upper section bonus	If the Sum of upper section is larger or equals to 63, you get 35 extra points in this category	35
Three-of-a-kind	Three dice showing the same face	Sum of all dice
Four-of-a-kind	Four dice showing the same face	Sum of all dice
Full house	Three dice showing the same face and two dice showing another face	25
Small Straight	Four sequential dice (Example, 2-3-4-5)	30
Large Straight	Five sequential dice (Example, 2-3-4-5-6)	40
Yahtzee	Five dice showing the same face	50
Chance	N/A	Sum of all dice
Total Score	N/A	The sum of the score in every category

Appendix B. A Detailed Comparison between this work's and Tom Verhoeff's algorithm.

Category	This work's algorithm	Tom Verhoeff's algorithm
Ones	1.35	1.82
Twos	4.28	5.25
Threes	6.83	8.57
Fours	9.91	12.19
Fives	12.99	15.74
Sixes	16.26	19.29
Bonus	5.25	24.14
Three-of-a-Kind	21.05	22.23
Four-of-a-Kind	10.41	13.04
Full House	20.64	22.86
Small Straight	29.91	29.53
Large Straight	35.21	33.04
Yahtzee	17.95	15.89
Chance	19.00	22.26
Grand Total	211.04	245.87