FRUIT SALAD

PROJECT 3

Programming and Problem Solving

Fall 2013 / Professor Ken Ross

GROUP 1

Ayushi Singhal

Enrique Cruz

Andrew Goldin

Pooja Kalpana Prakash

November 11, 2013

TABLE OF CONTENTS

1 INTRODUCTION

1.1 PROBLEM STATEMENT

The goal of this project was to create an agent (or player) that can intelligently play a game in which each player decides whether to keep or pass on a bowl of fruit served to them from a main serving bowl (or platter). Several players with various preferences for fruit are seated around a table, with the platter at one end. The platter contains a certain amount of 12 different types of fruit (types A through L). The total fruit in the platter can be represented by:

$$platterSize = n * bowlSize$$

Where *platterSize* is the total pieces of fruit in the platter, *n* is the number of players at the table and *bowlSize* is the total pieces of fruit in each bowl. Every bowl that is passed around contains the same number of pieces of fruit. Fruit is picked from the platter as follows: a particular kind of fruit (A through L) is chosen at random, with more plentiful fruits having a higher probability of being chosen. Then a value *c* is assigned as a random value between 1 and 3, and *c* pieces of the chosen fruit are added to the bowl. If there are less than *c* pieces of that fruit in the platter or the bowl requires less than *c* pieces of fruit to fill, *c* is accordingly reduced. A bowl is filled once it contains *bowlSize* pieces of fruit.

Each player's fruit preference is determined by a value assigned to that type of fruit. A fruit with a preference value of 12 is a player's favorite fruit, while a fruit with a value of 1 is a player's least favorite fruit. Each of the 12 different fruits is assigned a unique preference for a player. As such, the utility of a bowl can be scored by the following formula:

$$score = \sum_{i=1}^{12} p_i n_i$$

where i represents a certain type of fruit (letting fruit A = 1, B = 2, …, L = 12), p_i is the player's preference value for fruit i, and n_i is the amount of fruit i in the bowl.

The game is played as such: once the first player receives a bowl, (s)he will be given the choice to keep the bowl if it has a satisfactory score, or pass it on to the next player, at which point the second player will be tasked with the same choice, and so on. Once a player chooses a bowl, they must pass on any additional bowls they get. The last player at the table who has not chosen to keep a bowl is forced to take the first bowl (s)he gets, since there is no one left to pass to. This process continues until all players have chosen a bowl, at which point their scores are recorded. Then a second round is played in which the positions of the players at the table are reversed, with the last player from the previous round becoming the first player. The platter is reset, the process from the previous round is repeated and the scores of each players' two bowls are added for a

final score. A player is considered to have done well if their final score exceeds some expected value for the player dictated by the distribution of fruit in the platter.

In each turn of the game, every player is aware of the following: the total number of players, their own personal preferences, the bowl they receive and the type and number of fruit in it, the ID of the bowl (e.g. if the ID is 5, then it is the 5th bowl created in that round), and current round number (first or second round). The goal of the project was to use this information to create a player who would attempt to maximize their average score over many games by intelligently considering how to choose a bowl.

1.2 KEY TERMINOLOGY

The following are definitions of some terms used throughout the report, presented here for clarification:

Platter - The main serving bowl of the game, containing all of the fruit that will be distributed to each player.

Bowl - A bowl generated from the platter to be passed to the players. It contains a number of fruit equal to the size of the platter divided by the number of players.

Bowl size - The number of pieces of fruit in a bowl (*not* the physical size of a bowl).

Uniform distribution - A distribution of probabilities of the fruit in the platter, where each fruit has an equal probability of appearing.

Expected value - Our player's calculation of the average score of a bowl, based on an estimation of the distribution of fruit in the platter and our fruit preferences.

Player - Another term for the agent (in this case our program) that decides to keep or pass bowls of fruit.

Dumb player - The player given to each group at the start of the project. The dumb player implements a simple strategy that is suboptimal, choosing to keep a bowl with 0.5 probability.

Round - There are two rounds in the game, one where bowls are passed in one direction from the first player to the last, and a second where bowls are passed in the opposite direction, from the last player to the first.

Turn - A turn has passed when a single player chooses to keep or pass a bowl. A round ends when a sufficient number of turns have occurred such that each player has chosen to keep a bowl.

Tournament - A series of many different games, where each game has a unique configuration and a unique distribution, on which all players are tested.

Game - A unique configuration and a unique distribution which all players were tested against in the tournament.

Timeout - If a particular player throws an exception or exceeds the max allotted time by the simulator (50 ms) to make a decision, it is treated as a "timeout" and the player will continue to pass bowls for the remainder of the game. This is a situation that struck our player pretty significantly, and affected our overall performance compared to locally run tests without the timeout stipulation.

2 STRATEGY

2.1 OVERVIEW & EVOLUTION

Throughout the entire project, our player's primary heuristic revolved around attempting to estimate the distribution of the fruits in the initial platter in order to make our decisions. We began our strategy by assuming a uniform distribution and calculating the expected value of our score based on this distribution and our preferences. If the bowl shown to us had a higher score than our expected value we took it else we rejected it.

We improved upon this strategy by estimating the distribution based on the bowl shown to us. To do so, we first assumed that each fruit had an equal probability and updated the probabilities of each fruit to the number of that particular fruit we saw over the total number of fruits seen so far. These values were then normalised to ensure that the probabilities added up to 1. We emulated the simulator function of creating serving bowls based on the fruit distribution. However, this distribution was our estimate of the number of each fruit based on the probabilities calculated by us and then removing the number of times we have seen each fruit. This left us with the current platter from which the next serving bowl would be chosen.

By using the simulator's function of creating serving bowls, we accounted for the clustering effect as well. We created 10000 such bowls and calculated the average score of these bowls as our expected value. We initially calculated our max value as 12 * number of total fruits in a serving bowl. Then we performed linear interpolation to see how far away our expected value was from our maximum value attainable based on the number of bowls we would see in that particular round. After rigorous testing we discovered that maximum value was too high, which led our player to reject most of the good bowls as per our preferences. We changed the definition of the maximum score to be the highest score observed of all the 10000 bowls we generated. However

this was still to high as we noticed that the probability of getting at least one extremely large value in the 10000 bowls was very high. In order to address that we then proceeded to calculate the standard deviation from the distribution of our 10000 bowls. And performed a linear interpolation between our empirical EV and Υ=0.8 standard deviations away from in. All of the details of our players inference procedure as well as decision making boundaries will be explained in the following sections of this report.

This method worked well with uniform distributions and other distributions where most of the fruits were present. However, in distributions where most fruits were not present, our player did not perform well. This was due to the fact that we started from a uniform distribution and never modified our probability for a fruit if we did not see it even once. Hence, even though due to normalisation, the probability of this fruit dropped, the drop was too small and the probabilities were still too close to show much difference in estimating the number of fruits in the platter.

We wanted to penalize the absence of fruits but not too much since the fruits could be present in the platter and just be absent from the bowls we see or the fruits could have just been missing from the initial bowls we see. If these fruits were high in our priority and we assumed they were absent, it would lead to an incorrect estimation of distribution with lower score for us, leading us to accept average bowls. Hence after a lot of rigorous testing, we decided on two strategies for updating the probabilities of the fruits. Each strategy is implemented in one of our players and has been discussed in detail below.

2.2 INITIAL PLATTER ESTIMATION

As described in the initial overview, in order to make a decision on whether or not to keep a bowl, we start off by making an estimate of the initial platter. The more accurate the estimation, the better will be the choice of the bowl. For our final submission we had two approaches to make this estimation and both of these were submitted as two different players. The implementation of both of these players varied in the probability estimation part. Though both the players began with a uniform distribution, they differed from each other in the logic behind updating the expected probabilities of each of the 12 fruits.

VERSION 0 PLAYER

Our first version of the player updates the probability estimates as follows.
For each fruit type, if the number of fruits of that type (say type A) seen so far is greater than the number of fruits we would have seen if the distribution of the fruit was uniform, the probability of the fruit of type A is incremented by a factor. For example, if the bowl size is 24, the probability of each fruit is 12/24. This implies that the expected number of each fruit is 2. However, if we see 3 fruits of type A in the first bowl we receive, this implies that more fruit of this type exists. Hence, the probability of this fruit is increased. The probability is increased as follows:

$$\frac{(No.of\ type\ A\ fruits\ actually\ seen) - (No.of\ type\ A\ fruits\ expected\ to\ be\ seen\ under\ a\ uniform\ distribution)}{Total\ number\ of\ (all)\ fruits\ seen\ so\ far}$$

Alternatively, it is possible that only three fruits of type A existed in the platter, and all of them got selected at once due to the effect of clustering. To accommodate for this, we always implement the following check:

$$No\ of\ type\ A\ fruits\ actually\ seen > \frac{bowl\ size}{12} \times number\ of\ bowls\ seen\ till\ now$$

Hence if fruit A does not appear the second time, the number of type A fruits seen would still be 3. But now we would be comparing it against (24/12)*2. Hence the condition would not be satisfied. Since we start with uniform probabilities, and the above check implies the type A is present less than it would be in a uniform distribution, its probability is decremented by the following factor:

$$\frac{No.of\ bowls\ seen\ till\ the\ current\ round}{Total\ no.of\ fruits\ in\ the\ platter}$$

Note that these probabilities are only used to make an estimation of the platter. Each of the fruits seen so far are always decremented from the platter before making a decision on whether or not to keep a bowl.

VERSION 1 PLAYER

In this version, we first calculate the total fruit of a type (say A) seen as a fraction of the total number of fruits seen so far. This fraction can be mathematically represented as follows.

$$\frac{No.of\ type\ A\ fruits\ seen\ so\ far}{Total\ number\ of\ (all)\ fruits\ seen\ so\ far}$$

If this value is greater than 90% of the number of type A fruit we would have expected to see when the distribution of the fruits in the platter were uniform, we increase the expected probability of the fruit in the initial platter by a factor that is equal to the fractional representation of the fruit explained earlier. On the other hand, if the number is less than what is expected, the probability of the fruit is decremented by the same factor that was used in g1_v0.

Relation between probabilities of round 1 and round 2: We observed that our estimation of the platter improves with the number of bowls we see. If we are seated at a good position in round 1, then we could have gathered good data about the platter, which could be used in round 2 for our estimation. However, if we are seated at a poor position in round 1, our estimation will be slightly biased towards the fruits we saw in the limited serving bowls we were shown. In the worst case, if the serving bowls in the second round initially contain these fruits mostly, our estimation would become very highly biased towards these fruits. This would interfere with our expectation and could lead us to accept average bowls when we could have received better bowls at a later turn. To avoid this error in the computation, we determine, when it is optimal to carry

over the probabilities over to the next round and when it is not. After rigorous testing we found that if in round two our seating position at the table was greater than 1/3 * number of players then the probability estimation was quite accurate to the distribution. If this condition was not satisfied, then in the second round the probability estimations would begin with uniform probabilities.

2.3 DECISION BOUNDARY

Correctly inferring the distribution of the serving platter from the seen data only represents the first half of the problem, the player must correctly decided whether it believes the bowl it currently has is good enough to take it, forfeiting any following bowls that might be better, or to pass on the bowl, deciding instead that he believes the likelihood of a better bowl coming along in the future is high.

The general idea behind the strategy we chose to implement is the following. We rely on our probability inference to, through the mechanisms described in the previous sections of our report, estimate the original make up of the platter, then subtract the quantities of fruits we have seen and yield an approximation of the serving platter from which all subsequent bowls will be served. From here we are able to empirically calculate the EV for our player given this serving bowl by generating 10000 serving bowls mimicking the clustering distribution and fruit picking mechanisms as the game simulator.

From here, the simplest decision would be to take a bowl if it simply is higher than our expected value, but this lends itself into the following two problems:

- Because it is greedy, we take the first bowl we see that beats our expectation, this means that we overvalue early bowls. Even if we get a marginally better bowl than our expected value on the first move, and thus the chances of getting a better bowl further down the line are high would take it.
- We further overvalue early bowls by not taking into account the value of the information we would get from subsequent bowls. As we see more bowls our estimation for the EV is bound to get more accurate, and thus we would make better decisions.

This intuitively points to one conclusion. The more bowls one is guaranteed to be left to see, the stricter the decision criteria for taking a bowl should be. This guarantees that we take into account how good this bowl is given the likelihood of getting a better bowl further down the line as well as the value of acquiring more information before deciding to take a bowl.

To achieve this we developed a dynamic decision boundary that adjusts the threshold for taking or passing a bowl as a function of both the empirical distribution of our estimated platter and the number of bowls our player has left to see.

OVERVIEW

The fundamentals of our strategy consists of the following, defining a lower bound, an upper bound and a function of the bowls left to see that will be used to interpolate both bounds. Essentially when the match begins we define a function such that:

$$b(n) = \textit{Decision threshold} \quad \forall\, n \in [1,k] \quad s.t.:$$
$$k = \textit{total \# of bowls to see} \qquad b(1) = \textit{Lower Boundary} \qquad b(k) = \textit{Upper boundary}$$

IMPLEMENTATION

Lower Boundary (LB): Our decision thresholds lower boundary was quite easy to define, it is simply our empirically calculated EV. This means that when there is only one bowl left to be seen we will only take the current bowl if we expect it to be better than the only other one we will get to see which is clearly the optimal decision in this case as we are not sacrificing any further information and our decision simply becomes is the current bowl better than what we expect to get the next time.

Upper Boundary (UB): The upper was defined as EV + Υ*SD. This essentially means that the upper boundary is defined as a coefficient of the variance we see when we generate our empirical EV. This was calculated as:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} \qquad \text{where } \bar{x} \text{ is the empirical EV}$$

By making our upper threshold proportional to the variance we see in our empirical bowl distribution, we are adjusting for the likelihood of seeing a higher scoring bowl. By the definition of our upper threshold, a higher sample variance means that we have a higher threshold since there is a higher chance for bowls to be significantly higher than the EV, the opposite can be said for a small sample variance. The Υ value used for our final strategies was found by performing rigorous testing that will be explained in later sections of this report.

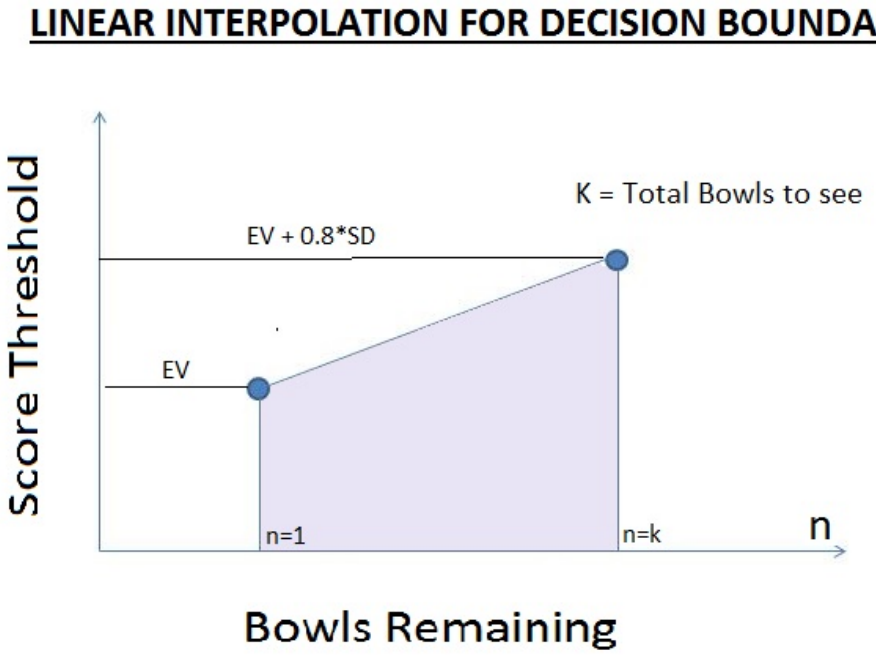
Interpolating Functions: This function essentially encapsulates our players value towards unseen information as well as his perception of the likelihood of getting a better bowl in the future given the number of choices he has left to make. In order to experiment with different interpolation rates from the upper to lower boundary we tested our player with the following two functions.

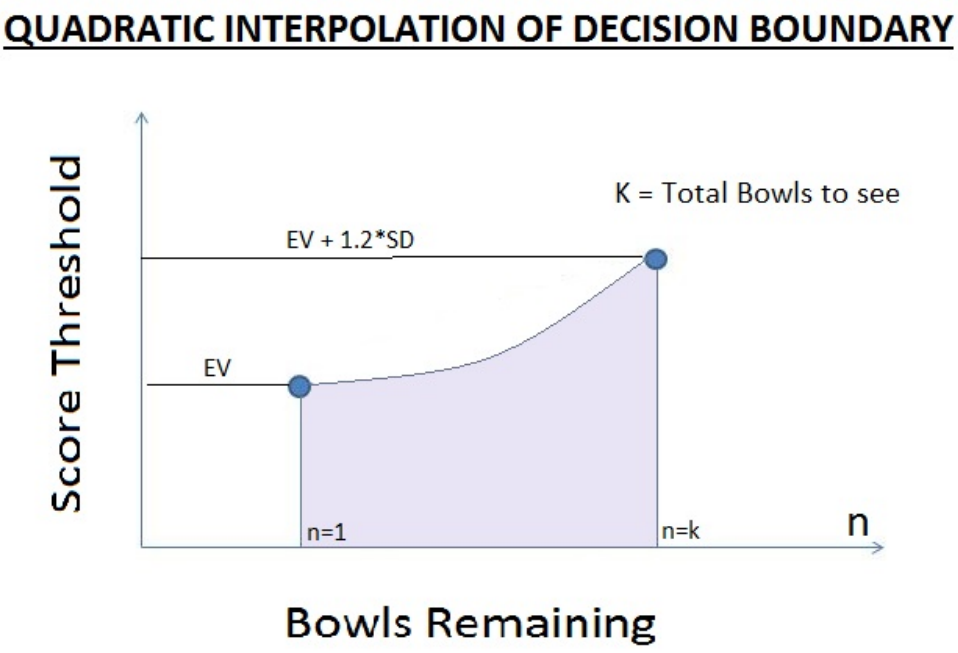
1. **Linear Interpolation:** We take the difference between the lower bound and upper bound and divide it equally into k segments, at each turn the boundary becomes the lower threshold plus n of this segments.
2. **Quadratic Interpolation:** We take the difference between the lower bound and upper bound and divide it into segments each of half the size the previous one; the first being of size (LB-UP)/2 and the last being the unassigned portion. At each turn the boundary

becomes the lower threshold plus the corresponding n segments.

FINAL DECISION STRATEGIES

After having defined our boundaries and the two functions to be used for interpolation we proceeded to do significant testing in order to determine the the optimal Υ value for each interpolation method (0.8 for linear and 1.2 for quadratic). Below are diagrams representing both of the final decision strategies that we tested our player on





In the end as will be shown in the results discussed later in this report it became clear that the linear interpolation function yielded better results, thus in our final submission for our player this was the function being used for our decision strategy.

3 JUSTIFICATION FOR STRATEGY

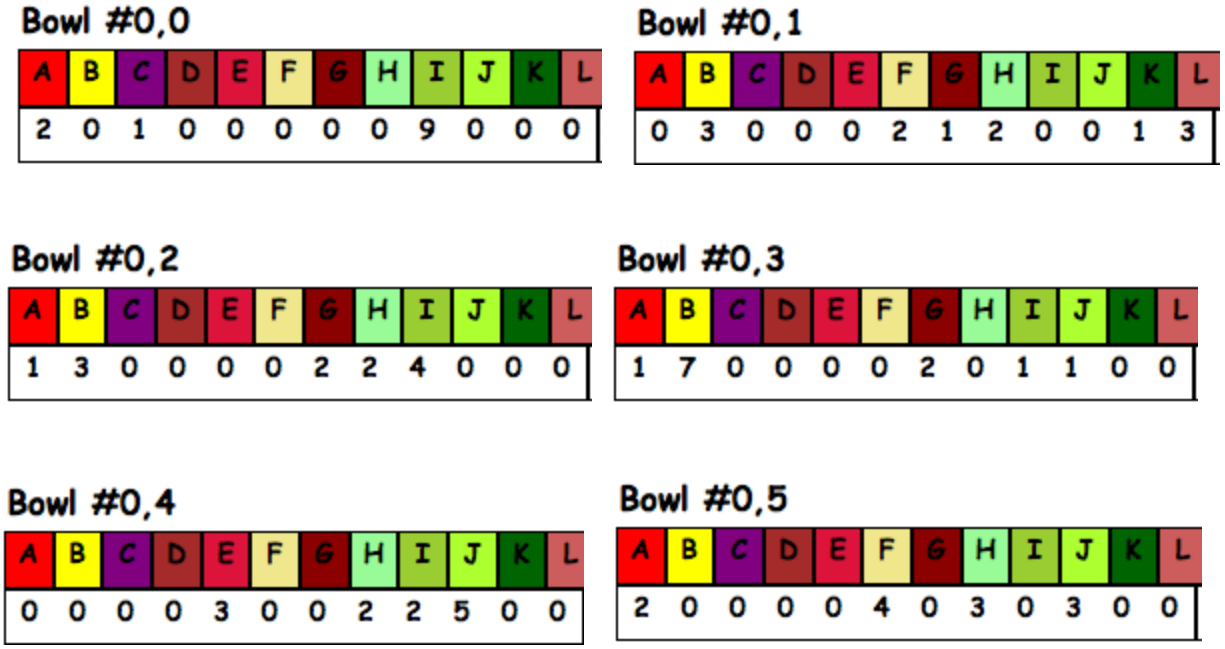
3.1 STARTING FROM UNIFORM DISTRIBUTION

We began our probability estimations assuming that the initial distribution of fruits in the platter was uniform. We did not begin by assuming 0 probability for each fruit since in this case, the probabilities of the fruits became highly biased towards the fruits we see and the quantity of each fruit we see. This might lead to inaccurate results since some fruits may have 0 probability in our distribution but could exist in a decent quantity in the actual platter. One such case is illustrated in the example below.

The initial distribution of the platter is as follows:

	A	B	C	D	E	F	G	H	I	J	K	L
I	6	19	5	0	16	14	10	15	36	14	5	4

The fruit distributions of the first 6 serving bowls have been shown below:



We can observe from the above data that fruit E appears only in one bowl though the total quantity of fruit E is present more than it would be in a uniformly distributed bowl. Now consider that player in position 1, accepts bowl 0,4 which was the only bowl that contained fruit E. In the event that player 2 starts with 0 probability of each fruit, it would assume that fruit E never occurs because it sees all the other 5 bowls which do not contain fruit E. If fruit E was the highest priority of player 2, its expected value will get reduced and hence it may end up accepting a bowl with a lower score even though it could have got a bowl with fruit E and, perhaps a higher score. However, if the strategy began with a uniform distribution for all the fruits then even though the probability of fruit E would be reduced in its absence in the bowls seen by player 2, it would most likely not be 0, and would hamper the expected value in a diminished manner.

3.2 PROBABILITY ESTIMATION

The initial probability distribution of the fruits begins with a uniform distribution. Whenever we receive a serving bowl, we check if the the cumulative sum (over all turns in each round) of each fruit we have seen till now are less, equal to or more than the number of that fruit we would have seen if the main serving platter consists of a uniform distribution of the fruits. If the cumulative sum of the fruit is equal to the uniform distribution value then the probability is unchanged, since it is exactly corresponding to a uniform distribution. If the sum is more than the uniform distribution value, then we calculate the current probability of the fruit based on the distribution we have seen and the probability of the fruit in a uniform distribution as follows:

$$probability\ in\ uniform\ distribution: \frac{Number\ of\ type\ A\ fruit\ expected\ to\ be\ seen\ under\ a\ uniform\ distribution}{Total\ number\ of\ all\ fruits\ seen\ so\ far}$$

$$probability\ in\ distribution\ observed: \frac{Number\ of\ type\ A\ fruit\ actually\ seen}{Total\ number\ of\ all\ fruits\ seen\ so\ far}$$

$$probability\ \Delta = probability\ in\ distribution\ observed - probability\ in\ uniform\ distribution$$

The new delta probability calculated above shows how far away the current probability of fruit A is from the initial assumption of a uniform distribution and this is corrected by adding it to the initial uniform distribution probability. In essence, this equation simplifies to simply adding the probability in the distribution observed. However we have shown the above formulae to explain how and why we achieved our final probability estimate.

Now if the number of type A fruit in the distribution observed by our player is less than the number of type A fruit in the uniform distribution then the probability estimate of type A fruit is reduced. However, it is not reduced by a factor which is equal to the difference in the probability of type A fruit in the distribution observed and probability in uniform distribution. If we had implemented this, then once a fruit was not seen its estimated probability would reduce to 0, and the probability estimates would become similar to starting from a 0 probability distribution and updating the probabilities for fruits seen. Hence, we penalised the deviation from the uniform distribution by a small amount which would neither penalize too little, nor too much. After trying several factors for decrement, the following factor was deemed efficient:

$$\frac{1}{total\ number\ of fruits\ in\ the\ platter}$$

This factor was multiplied by the number of bowls seen by our player so far, because we observed that when we had more information, these comparisons became more representative of the distribution of the serving platter and hence we needed the penalization factor to be larger.

The probability of a fruit in a platter is:

$$probability\ of\ type\ A\ fruit: \frac{Total\ number\ of\ type\ A\ fruit}{Total\ number\ of\ all\ fruits}$$

In a uniform distribution this number is fixed at the constant value 1/12 since there are 12 types of fruits. Hence when a fruit appears less than the uniform distribution in the first bowl we

decrement its count in the entire platter by 1:

$$probability\ of\ type\ A\ fruit: \frac{Total\ number\ of\ type\ A\ fruit - 1}{Total\ number\ of\ all\ fruits}$$

After we have seen x bowls, if the fruit appears less than the uniform distribution for x bowls, we decrement is probability x times:

$$probability\ of\ type\ A\ fruit: \frac{Total\ number\ of\ type\ A\ fruit - x}{Total\ number\ of\ all\ fruits}$$

Finally, after all the probabilities were calculated the value of each probability was divided by the total sum of probabilities. This was done to normalize the values to ensure that the total sum of the probabilities did not exceed 1.

The player g1_v1 differs from g1_v0 in the condition to update and decrement the probabilities of a type of fruit. In g1_v0 we directly check if the number of fruits in the observed distribution is greater than the number of fruits according to uniform distribution at that point. In the g1_v1 however, we relax the condition a bit and check if the number of fruits in the observed distribution is greater than 90% of the number of fruits according to the uniform distribution at that point. This was done because we observed that at times, some of the fruits values may be very close to the uniform distribution, but not equal to or greater than. In this case decrementing its probability could reduce its probability more than necessary. The value 90% was chosen such that the constraint would not be too relaxed and not too strict. We ran several tests for the same player at the top position for different configurations and compared how well the probability estimation of the distribution was compared to the actual estimation of the platter. Since it is not possible to show the results of all such tests in the report, we have shown one particular case below.

Distribution: random.txt; Total number of players=12; Bowl size=12; Player position in table=1

Constraint value	A	B	C	D	E	F	G	H	I	J	K	L
80%	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333	0.08333333
90%	0.055276	0.055276	0.055276	0.055276	0.180905	0.060302	0.055276	0.120603	0.180905	0.060302	0.060302	0.060302

The above table records the findings when we run tests for 80% and 90% as the constraint value. At 80%, the player calculates the distribution probabilities as a uniform one. At 90%, the constraints are more strict than at 80% and the probability estimations are closer to the actual distributions of the platter.

3.3 CHOICE OF ⅓ COEFFICIENT

We performed various experiments in order to determine which fraction of the seating arrangement in the second round would lead to a good situation in which we could possibly conclude that the probability estimates of each fruit in the initial platter made in the first round is worth being passed on to the second round. We chose the fraction ⅓ which essentially means that if the we are seated in the top ⅔ rd positions in the first round, our probability estimation of the platter could be deemed close enough to the actual platter used by the simulator. To show the effectiveness of this value, we can see the following examples.

Distribution: uniform.txt; Total number of players: 12; Bowl size: 12

Player	Position in round 2	Carry over?	A	B	C	D	E	F	G	H	I	J	K	L
p1	2	no	0.047414	0.155172	0.051724	0.047414	0.047414	0.155172	0.047414	0.047414	0.047414	0.047414	0.047414	0.258621
p2	8	yes	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333	0.833333

The carry-over value in the third column indicates whether or not the player chose to carry over the estimates from the first round to the second. We can see in the above table that p2's (which does not carry over the estimate) estimation of probabilities at the end of the first round is accurate, where-as that of p1's is quite far away from the actual distribution. Hence carrying over p1's estimation would lead to the assumption that fruits B, F and L dominate the platter, however that is not true since the actual distribution is a uniform distribution.

Distribution: random.txt; Total number of players: 26; Bowl size: 36;

Player	Position in round 2	Carry over?	A	B	C	D	E	F	G	H	I	J	K	L
p1	20	yes	0.05149	0.0971	0.05149	0.05149	0.126023	0.121891	0.05149	0.104331	0.158045	0.083671	0.05149	0.05149
p2	8	no	0.055158	0.074499	0.055158	0.055158	0.223496	0.055158	0.055158	0.055158	0.111748	0.148997	0.055158	0.055158
random.txt	-	-	0.04347826	0.0973913	0.02782609	0.01391304	0.14086957	0.14956522	0.07304348	0.11652174	0.16869565	0.10086957	0.03652174	0.03130435
p1's difference	-	-	0.00801161	0.00029158	0.02366378	0.03757683	0.01484652	0.02767408	0.02155361	0.01219119	0.01065036	0.01719853	0.01496813	0.02018552
p2's difference	-	-	0.01167933	0.02289274	0.02733151	0.04124455	0.08262614	0.09440762	0.01788589	0.06136415	0.0569478	0.04812757	0.01863585	0.02385325

In this setting, p2's position just missed the cut off for carrying over the probabilities. p1's probabilities are carried over to the next round and the probabilities shown are the estimated probabilities of each fruit in the distribution at the end of the first round. We calculate the absolute difference of each estimated probability for p1 and p2 from the true distribution. We

observe that the differences are very high for p2 as compared to p1. Hence we can see that in the border case, ⅓ is a good coefficient to determine if we should carry over the probabilities from round 1 to round 2, based on the seating arrangement of the player.

Also, we would like to point out that as per the class discussions we learnt that a meaningful game would require at least 3 players. Hence in the first round, players in position one and two would have meaningful information about the platter distribution while the player in position three might not have enough data as it can base its estimate only on the one bowl it has seen. Hence our 1/3 coefficient is satisfied in the most base case and the probabilities of players in position 1 and position 2 can be generally be safely carried forward to the next round.

3.4 CHOICE OF SAMPLE SIZE FOR EMPIRICAL DISTRIBUTION

When we make the choice of whether to take a bowl or pass it on we must have first calculated what we expect subsequent bowls to look like in order to better judge if it is in our best interest to take it or not. To do so we decided to generate our own distribution of possible bowls to come; once we had inferred the initial makeup of the serving platter and subtracted the counts of elements we had seen we proceeded to generate 10000 bowls using the exact same fruit picking and clustering effects as the actual simulator (thus ensuring that we were mimicking the characteristics of the problem as close as possible). Similarly generating our own distribution had the added benefit of allowing us to identify the variability in the possible data to come and factor that in when making our decision. We found the sample distribution to be a normal one, and that it took approximately 10000 bowls to guarantee convergence (or close enough to it) for both the empirical EV as well as the SD, hence this was the value we used for generating our sample distribution in the final submission of our player.

3.5 CHOICE OF b(n) AND ϒ

Once we had settled on the general decision strategy that was to be implemented we proceeded to do some rigorous testing in order to determine both the ideal ϒ value to define our upper bound as well as which of our hypothesized interpolation function yielded better results. In order to do so a script was written to execute 2000 runs (value at which we found convergence to be almost absolute while still allowing for the testing to be tractable in terms of time) and test different ϒ values for each interpolation function. The script executed 2000 runs each on 3 different distributions, holding N_players and Bowl_size constant, then the average result obtained by our player in these 3 distributions was used as our performance metric.

Below are plots of the results obtained for different ϒ values for each of the functions:

It is easy to identify the maximum ϒ for each function, which in our final strategies were set to:

- **Linear ϒ = 0.8**
- **Quadratic ϒ = 1.2**

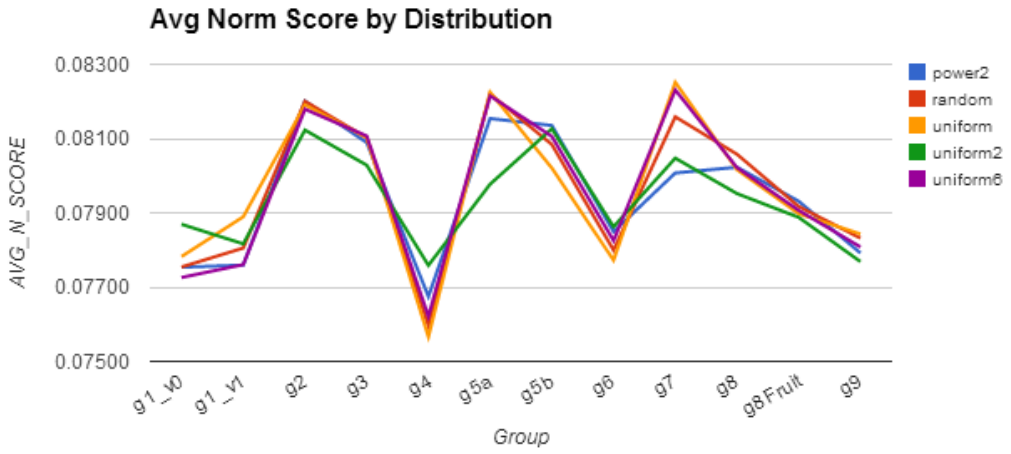
After having identified the optimal ϒ for each b(n) meth method we then proceeded to similarly conduct some testing to compare the performance of both methods, this time also introducing some variability in terms of the number of players and the bowl size. Although both methods performed very similarly it was found that the linear interpolation method yielded slightly higher results, thus in the final implementation of our player we chose **b(n)=linear** along with its optimal value of **ϒ =0.8**.

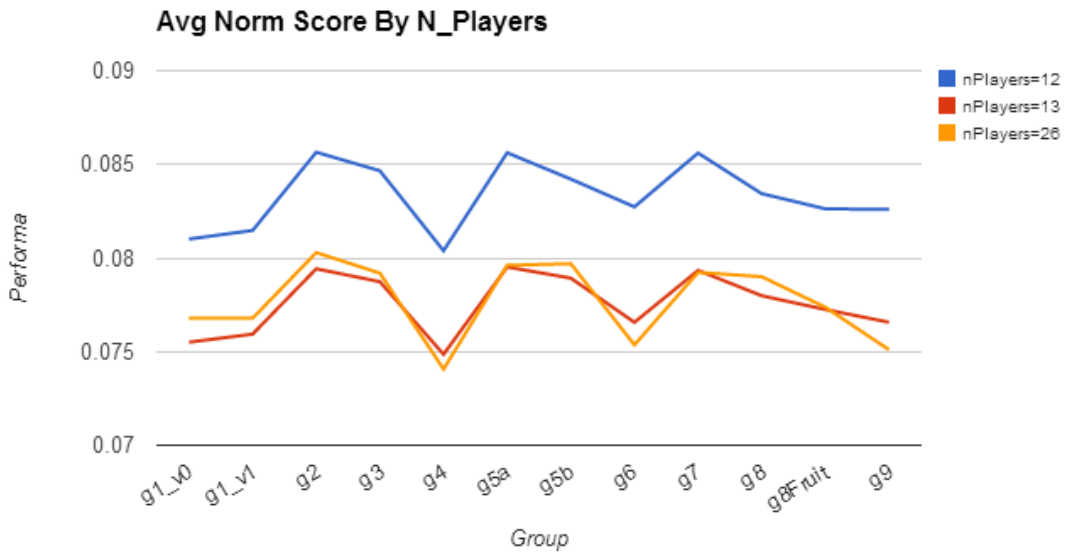
4 TOURNAMENT RESULTS ANALYSIS

4.1 OBSERVATIONS OF OVERALL RESULTS

WHERE WE STAND

In general, the performance of both of our players was highly varied, and seems to change quite a bit across different configurations. To visualize the performance of our players, we took the average scores of each group in every configuration, and for each configuration, normalized the scores to a 0-1 range by dividing each by the total of the average scores of all players in that game, and graphed the results. The following three graphs show the average normalized score for each player across the entire tournament, grouped by platter distribution, bowl size, and number of players.





There are a number of interesting points that we can glean from these graphs. We can see from the first and third graphs, representing normalized score by distribution and number of players, respectively, that variations in distribution and number of players don't have a strong of an effect on relative average player performance as do variations in bowl size. We can see from the graph displaying performance by bowl size that several players see substantial changes in performance as bowl size changes.

For some players, including both of our players and Group 4's player, average performance appears to exert a general upward trend as the size of the bowl goes up. This would translate to a general increase in average score with an increase in bowl size, likely with all other factors held constant. Conversely, groups 2 and 5 seem to suffer more when bowl size gets larger. In the individual analyses of our players, we will try to see if our performance in a select set of games reflects the data trends we see in these charts.

As another metric to visualize overall performance, the graphs below show the performance of both of our players, g1_v0 and g1_v1, as their scores relate to the highest and lowest scores achieved across each game run in the tournament. The solid lines represent the distance from each of our players' average scores to the score of the highest ranked player of that game, while the dotted lines represent the distance from the lowest score. Essentially, our players perform well when the dotted line is above the solid line, and relatively poorly otherwise. We can see that the two graphs are very similar, noting that our two players had similar performance.

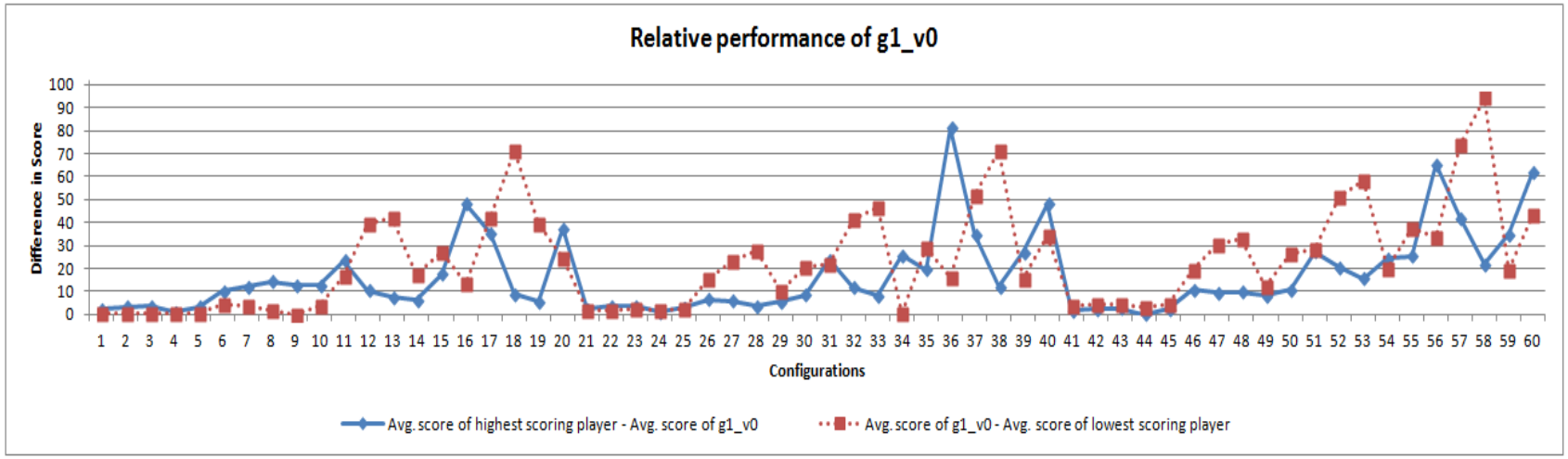
Looking at where the dotted lines fall above the solid ones, we can observe the game configurations in which we seem to have done relatively well. Looking at the charts, we see the dotted line falls clearly above the solid line for both players in games: 12, 13, 18, 27, 28, 32, 33, 38,

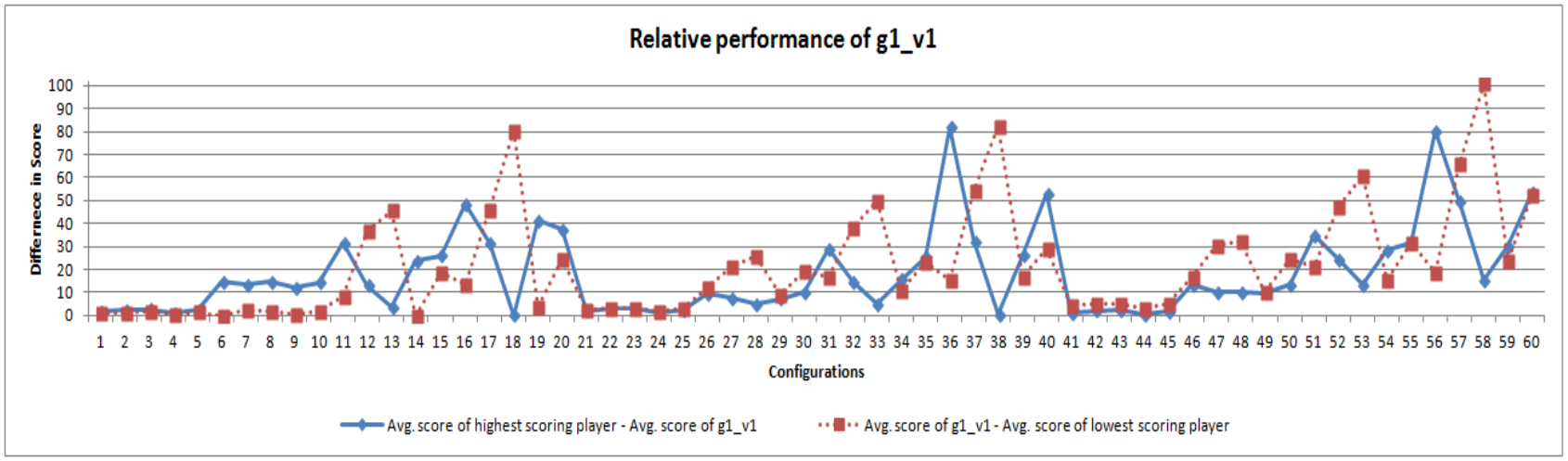
47, 48, 52, 53, 57, and 58, which correspond to the configurations in the following chart:

Game #	***Number of players***	***Bowl size***	***Distribution***
12	12	36	random.txt
13	12	36	uniform.txt
18	12	100	uniform.txt
27	13	12	random.txt
28	13	12	uniform.txt
32	13	36	random.txt
33	13	36	uniform.txt
38	13	100	uniform.txt
47	26	12	random.txt
48	26	12	uniform.txt
52	26	36	random.txt
53	26	36	uniform.txt
57	26	100	random.txt
58	26	100	uniform.txt

Games in which g1_v0 and g1_v1 have average scores which are closer to the highest score of that game than the lowest score

Another observation worth noting is that the farther apart the two lines are for any game, the higher the range of scores in that game, and the higher the variance. For example, game 44 has the configuration of 26 players, bowlsize 1, and distribution of uniform2.txt (two fruits with uniform distribution and 10 with probability zero). As discussed in the next section, Our v0 player actually had the highest average score on this game, but the graphs above do not capture this result well. This is because the variance of that particular game was quite low as compared with several other games played.

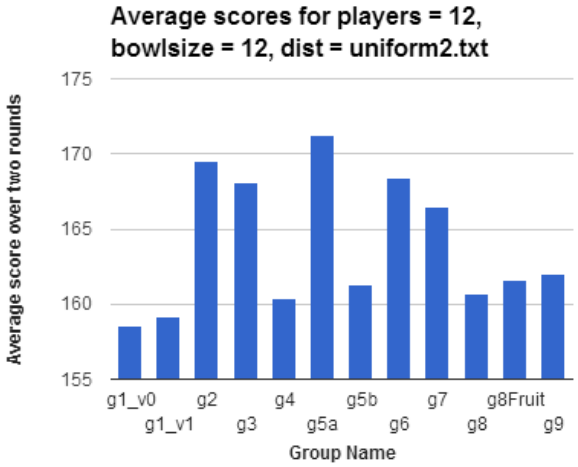

Relative performance of g1_v0
Difference in Score
Configurations
Avg. score of highest scoring player - Avg. score of g1_v0
Avg. score of g1_v0 - Avg. score of lowest scoring player


Relative performance of g1_v1
Difference in Score
Configurations
Avg. score of highest scoring player - Avg. score of g1_v1
Avg. score of g1_v1 - Avg. score of lowest scoring player

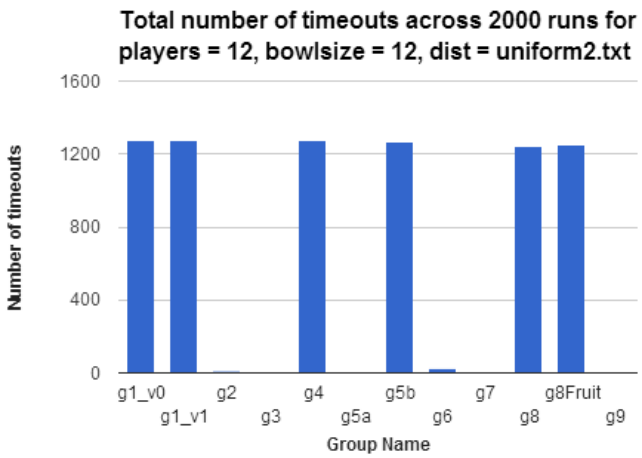
POOR PERFORMANCE AND SIMULATOR TIMEOUTS

After querying the tournament results, we discovered that for every 12-player game and several 13-player games, both of our players experienced a high frequency of timeouts. That is to say, for 2000 runs on each configuration, our player experienced timeouts on anywhere from 20% to 60% of these runs. When a timeout occurs for a player (meaning they exceed 50 ms of computation time), they will spend the rest of the current game only passing bowls. We hypothesize that these timeouts are one of the biggest contributors to our lackluster performance in the games with smaller player size. The most obvious culprit in our timeout situation is likely the method in our player that computes the expected value of a bowl, which runs 10000 times to generate its expected value.

As an example, the following graph shows the average scores of each player under the following configuration: 12 players (one of each group), a bowl size of 12, and a distribution in which two fruits have a uniform distribution and the other ten fruits have probability zero.



Our v0 player underperformed all other players, only garnering an average score of 158.5745. Something interesting to note about this particular game is that 9 out of the 12 players in this game experienced at least one simulator timeout. Observe the graph below:



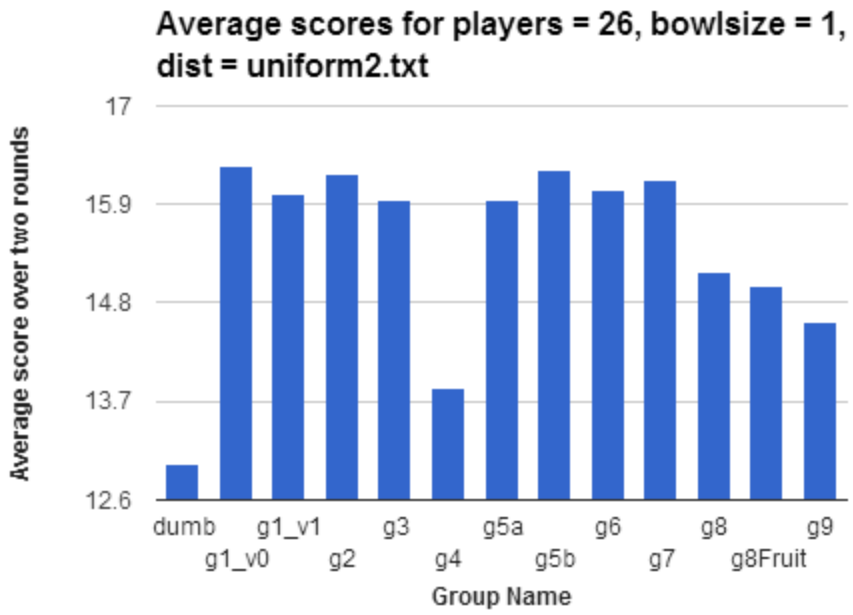
If we analyze these two graphs side by side, we see a fairly prominent correlation. Groups who experienced many simulator timeouts garnered much lower average scores in this game, while groups who did not experience timeout performed average or very well. In fact, our results show that in every game in which our players achieved the lowest average score, our players (and sometimes several other players) experienced a substantial number of simulator timeouts. We will look at more examples in the following section, accompanied by changes in performance when run on a simulator that does not give a penalty for timeouts.

4.2 ANALYSIS OF OUR PLAYER'S PERFORMANCE

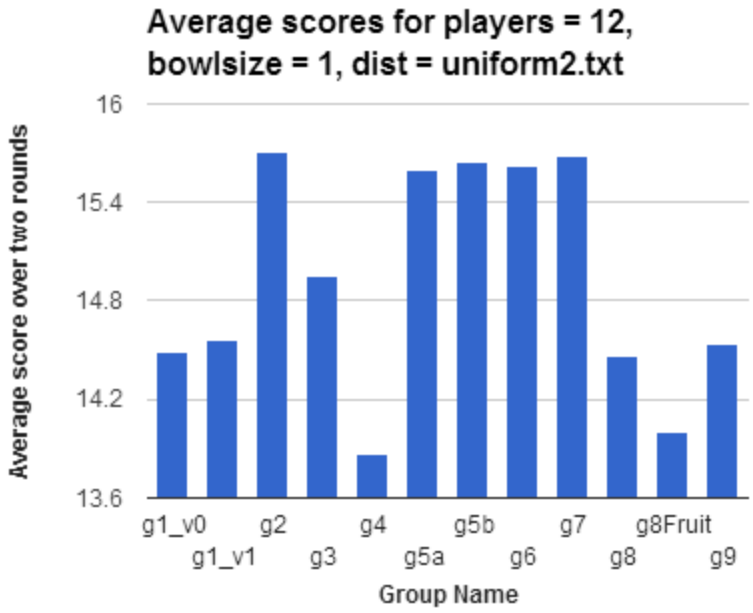
In this section, we will observe each of our players in two situations: games in which each player scores very well, and games in which each player scores poorly.

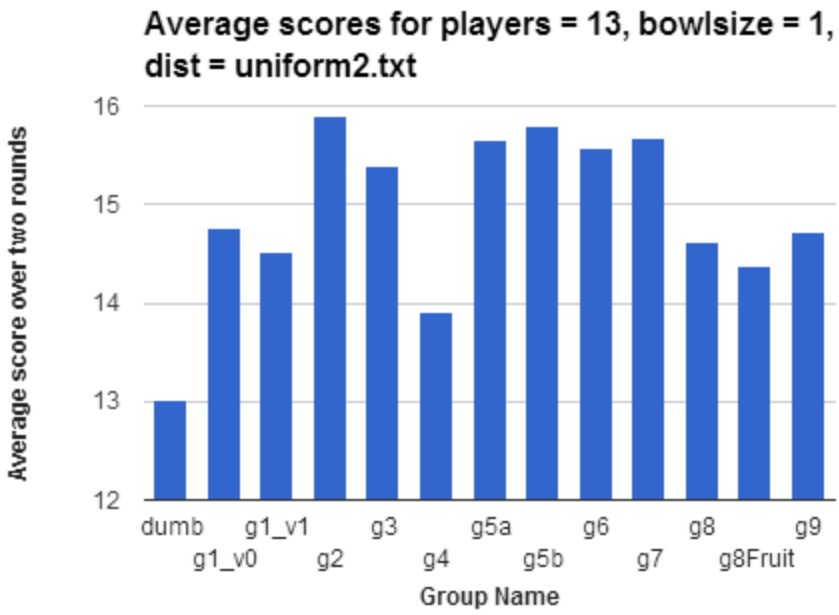
VERSION 0 PLAYER - TOP PERFORMANCE

The graph below shows the average scores of each player under the following configuration: 26 players (two of each group and two dumb players), a bowl size of 1, and a distribution in which two fruits have a uniform distribution and the other ten fruits have probability zero.



Our v0 player outperformed the other groups with an average score of 16.3305, followed by g5b, with an average score of 16.2843. It's somewhat unclear to see why our player performs well in this scenario until it is compared with results from similar configurations with fewer players. Observe the graphs below. They show the results with the same bowl size and configuration, but with a fewer number of players (12 and 13, respectively):

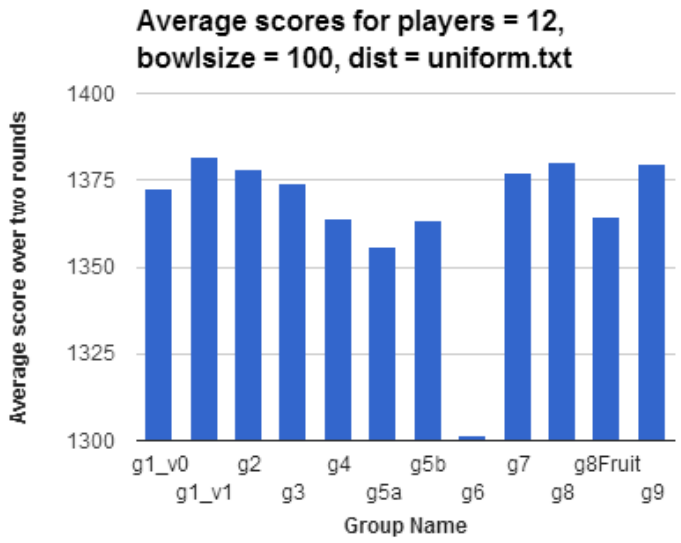




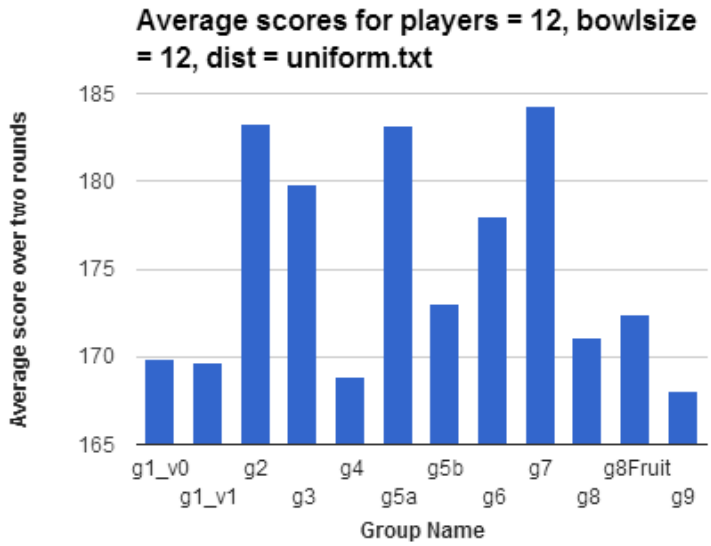
With 12 players, it can be observed that our performance for both players is greatly reduced, with v0 only garnering an average score of 14.489. With 13 players, v0's performance improves slightly, becoming 14.7615. Since our player relies on seeing a larger number of bowls to make an accurate estimation of the platter distribution, it stands to reason that having a larger number of players in the game will give our player the opportunity to see more bowls that pass by, and form a more accurate estimation of the platter. Subsequently, the player can make a more accurate determination about the expected distribution of fruits in bowls it has yet to see, and can better decide whether to keep or pass the current bowl.

VERSION 1 PLAYER - TOP PERFORMANCE

The graph below shows the average scores of each player under the following configuration: 12 players (one of each group), a bowl size of 100, and a distribution in which all fruits have a uniform distribution. Our version 1 player came out on top with an average score of 1381.7535, followed by Group 8 at 1380.4275.



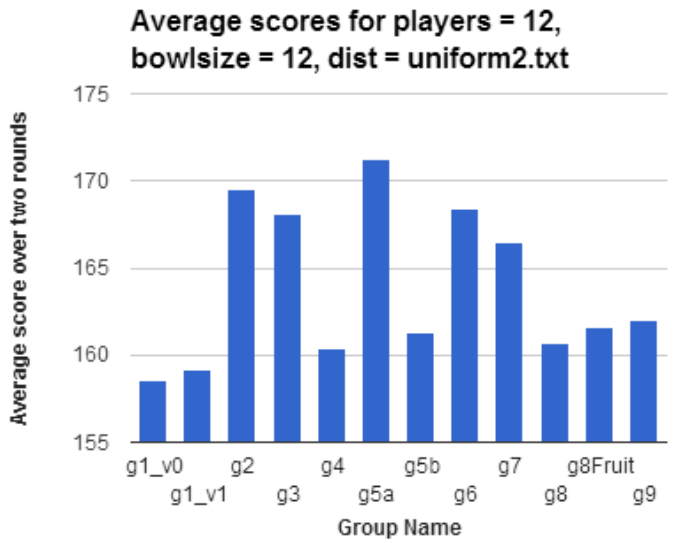
While it's refreshing in itself to see another top-ranking score from one of our players, the result becomes even more special when we observe a similar game configuration, with the bowlsize significantly reduced. The graph below shows the average scores for all players in a game with 12 players, a bowlsize of 12, and the same uniform probability distribution. The relative performance of g1_v1 decreases dramatically, now only ranking third from the bottom.



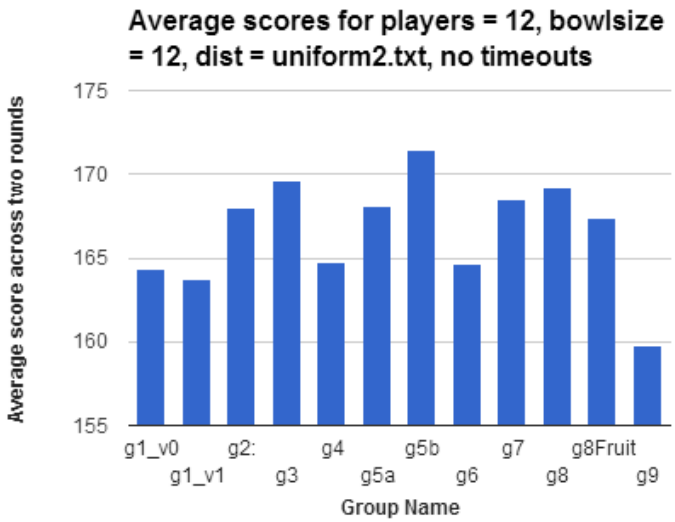
This observation further supports our group's hypothesis that the performance of both of our players seems to improve proportionally with bowl size and the number of players, as a result of having a greater wealth of information available to estimate the distribution of the platter.

VERSION 0 PLAYER - POOR PERFORMANCE

Going back to the following graph we observed in the section 4.1:



After running this configuration again without the timeout penalties from half of the players, we get the following graph:

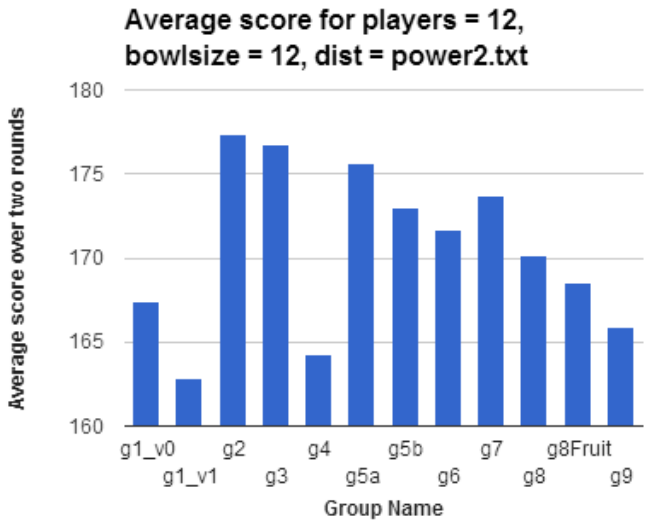


While the relative performance of our v0 player does not improve by much, the absolute score jumps by almost 6 points from 158.5745 to 164.347. Additionally, the absolute scores of group 5a and group 6 are decreased, while the scores of groups 5b and 8 increase substantially. Not only does the absence of the timing penalty seem to lower the overall variance in the scores, it also appears to affect the scores of players who did not experience timeout.

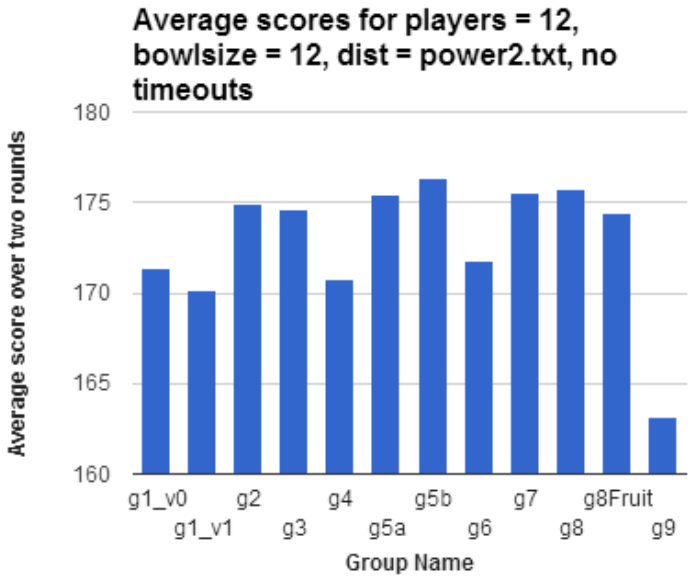
VERSION 1 PLAYER - POOR PERFORMANCE

The following graph shows the average scores of each player under the following configuration: 12 players, a bowl size of 12, and an exponential distribution in which the probability of each

successive fruit is halved. Our version 1 player made it through this game with only an average score of 162.8055, a difference of almost 15 points when compared to group 2's score of 177.3405.



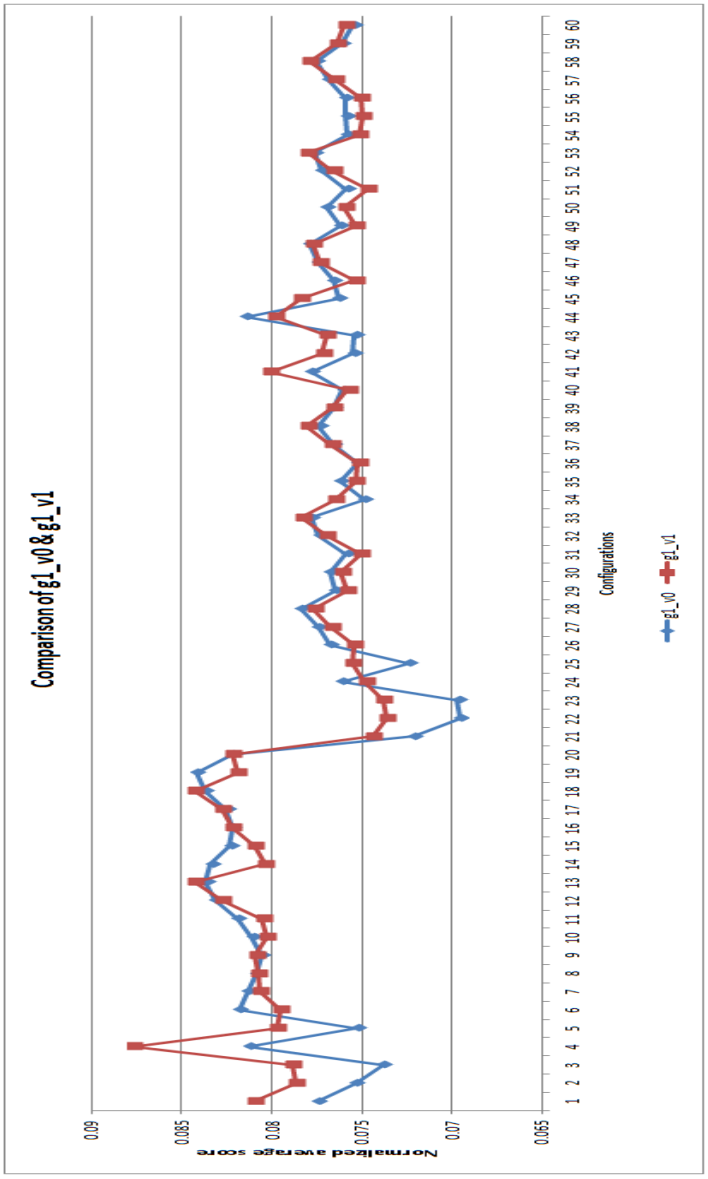
In this configuration, g1_v1 experienced timeout in 843 out of 2000 runs. After running the configuration again without time penalties, we obtain the following graph:



While the relative score of our player does not see much of an improvement, g1_v1's absolute score jumps almost 8 points to 170.1230. Additionally, the absolute scores of groups 2 and 3 are reduced while the scores of group 5, 7, and 8 are increased. The abolishing of the timeout penalty seems to have decreased the variance in scores by quite a bit.

DIFFERENCE BETWEEN VERSION 0 PLAYER AND VERSION 1 PLAYER

As can be observed from the graph below, the performances of both the version 0 and version 1 players is almost similar. Hence the relaxation constraint in version 1 player might give better results in some configurations as compared to the version 0 player, however this difference is offset by the better performance of version 0 player in some configurations as compared to the version 1 player.



5 FUTURE IMPROVEMENTS

We discovered that our code took more than 50ms to run which results in a lot of timeouts for our player. We could reduce the timeout in the future by reducing the number of bowls we generate to sample bowls from the estimated distribution of fruits. Since we did not have enough time to re run the tournament for all the configurations where we timed and properly analysed how your score might have been affected by the time outs, we feel we were unable to fully highlight the performance of our player without the CPU time restrictions.

We could also improve the distribution of fruits our player estimates based on the probability distributions by emulating how the simulator uses the probability distributions to create a platter. Currently we multiply the estimated probability distributions by the total number of fruits in the platter to obtain the estimated distribution. Hence for a uniform distribution the estimated probabilities would be uniform and multiplying the probabilities with the total number of fruits would give an estimated distribution with equal numbers of each fruit. But we observe that for a uniform distribution, the probabilities of each fruit is the same, however the actual number of each fruit in the initial platter created by the simulator is not equal. These numbers vary. Hence we could use the method employed by the simulator to calculate the estimated distribution based on the probabilities as it would account for the slight variation that the simulator creates and would help our player better emulate the platter used by the simulator.

Our player decreases the probability of fruits that appear less number of times than they would in a uniform distribution. Along with this we could derive some metric where it would be safe to assume that a fruit does not exist on the platter based on the bowl size, the number of bowls our player has seen and the number of players. This would improve our estimation further and could help us increase our score.

6 CONCLUSION

As previously mentioned, the underlying strategies of both of our players were twofold: (1) estimating the probability distribution of the serving platter and (2) using that distribution to generate an expected value for a bowl based on an interpolation heuristic, in an effort to intelligently determine which bowls of fruit are statistically better for our player to take. While the steps we took to arrive at our final strategy seemed logically sound, ultimately in practice neither of our players performed as up to our expectations. This was due to a combination of factors. Overall trends shows that our players performed slightly worse with small player numbers and small bowl sizes. Additionally, a great deal of the penalty we accrued throughout the tournament was due to simulator timeouts, in which our player failed to pick a bowl within the allotted turn time of 50 ms for each player in the game. We submitted two players primarily as a test to see which of the slightly differing approaches to our strategy was most effective, but ultimately the results for each player seem to indicate a wash, as the overall performance trends for each are

almost identical, only differing very slightly under certain conditions. Given extra time, we would have liked to have been able to run a more extensive series of head to head tests or stick with one implementation to focus on and refine for better overall performance.

7 APPENDIX

7.1 INDIVIDUAL CONTRIBUTIONS

AYUSHI
For the player, supplied most of the functionality for estimating the probability distribution of the platter. For the report, contributed significantly to the strategy evolution and strategy justification sections, particularly for justifying probability estimations and also the future improvements..

ENRIQUE
For the player, provided the final expected value, decision boundary and interpolation strategies for ultimately deciding to accept bowls. For the report, contributed to the strategy section on interpolation, and aided with data analysis for the tournament.

ANDREW
Supplied the initial expected value functionality and methods for estimating platter distribution to the player. For the report, analyzed and compiled the tournament results with respect to overall performance and timeouts, as well as provided the problem statement and key terms.

POOJA
For the player, worked with Ayushi to develop the functionality for estimating probability distribution. For the report, discussed platter estimation and justification, and discussed the reasoning behind creating and submitting two players.

7.2 ACKNOWLEDGEMENTS

We would like to thank Jiacheng for implementing and improving the simulator throughout the duration of the project and for running the official tournament. We would also like to thank Professor Ross for creating this complex and interesting project and for inciting thought-provoking class discussion. We thank the other groups for creating and improving their players throughout the project so that we could learn from their successes and failures and test their players against our own. We'd especially like to thank Group 6, who provided us with tools to help us retest certain tournament configurations more quickly.