UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG | SCHOOL OF ELECTRICAL AND INFORMATION ENGINEERING

Applied Quantum Computing

Course Project

ELEN4022 | Iverson Mpano (2320484) | Brayden Hand (2194051) | 30/03/2023

Abstract

This report presents a simulation of a simplified version of the dice game 'Craps' through classical and quantum computing approaches. The classical approach uses a pseudorandom algorithm to generate a 'random' number and analyses the unequal probabilities of the thirty-six possible outcomes to determine 'The Shooter's' chances of winning. In contrast, the two methods of the quantum approach utilize qubits to represent the dice, but the prime factors of six make it challenging to create a uniform distribution of the six possible outcomes. The report concludes by comparing and contrasting the two approaches and their limitations. It then concludes the classical approach outperforms the quantum approach.

1. Introduction & Background

The two dice game chosen is a simplified version of the game 'Craps'. The game functions by having a player, referred to as 'The Shooter' roll a pair of dice, whose sum determines what's known as 'The Point'. 'The Shooter' must now continue to roll the pair of dice until their sum is equal to 'The Point', in which case they win. However, if they roll a seven any time before then, they lose (SYCUAN, 2020). The game is then seen as 'weighted' towards 'The Shooter' losing. This is proved later on because when summing the die, it is seen that seven has the highest probability of occurring.

The most obvious constraints then of 'Craps' is that then 'The Point' can never be a seven a one or a number above twelve, and very low and very high numbers have a low probability of occurring. This is not an issue in the classical implementation but does prove to be more difficult in a quantum context. Regardless, code is included (in the git repository) plotting the distribution of dice rolls including seven. The largest constraint of the classical approach taken is the use of pseudorandom instead of true random number generators. This is explained further on. The success of the project relies on being able to accurately simulate the classical and quantum approach with suitable reasoning to their contrast.

The remainder of this report details approaching the simulation of this game via a classical and quantum approach. Comparing them and highlighting their differences to improve a software developer's understanding of the current limitations of quantum computing when investigating probabilistic problems with such things as two, six-sided dice. The report also investigates Einstein's comment, "God does not play dice with the universe" in line with what was learnt from comparing approaches.

2. Classical Approach

In classical computing, a software developer understands the concept of how random number functions rely on deterministic, pseudorandom algorithms to generate a sequence of numbers. They are not truly random and rely on an initial seed value, typically generated by the computer system's clock. This is a disadvantage compared to true random number generators which rely on physical sources of randomness (beyond the scope of this project) or quantum random number generators (Rohatgi, 2015). The Python 'random.randint()' function (pseudorandom), imported from the 'random' library is used in the classical approach to generate a number between one and six (Python Software Foundation, 2021). Hence, in the classical approach this function is used twice to generate a number per six-sided dice. The numbers generated are summed and this becomes 'The Point'. The code uses a boolean flag system to keep generating a new 'Point' to ensure it is never a seven.

We are working with two six-sided dice and hence understand there are thirty-six possible outcomes. However, these outcomes do not have equal probabilities (Calculated in 'Project_Classical_Approach.ipynb' and visualised in Figure 1). These unequal probabilities play an important role in the odds of 'The Shooter' winning. The losing factor being 'The Shooter' rolling a combined number of seven has been carefully chosen because there are six ways of making a seven from two dice. The largest number of ways among the other outcomes. 'The Shooter' then has a higher probability of losing because they are often expected to roll a seven before reaching 'The Point'. This is visualised in Figure 2 (and calculated as well in 'Project_Classical_Approach.ipynb').

The classical approach we have undertaken may not benefit from true randomness but is sufficient in this application. It is largely more dependent on the unequal probabilities determined by the physical attributes of the dice. However, the quantum approach may be able to provide the benefits of true randomness.

3. Quantum Approach

In quantum computing, qubits are used in the place of regular bits. Qubits are the fundamental building blocks of quantum computers and quantum information processing. Unlike classical bits, which can

only be in a state of 0 or 1, qubits can be in a state of 0, 1 or a superposition of both states simultaneously. Qubits can also become entangled, meaning that their states become correlated in a way that cannot be explained by classical physics. The state of a qubit is usually represented by a 2x1 vector ($\begin{bmatrix}1\\0\end{bmatrix}$) or a ket ($|0\rangle$) containing its current state. The state of a system of n qubits is represented by a vector with 2^n entries or as a ket containing the state of each qubit as one e.g., $|1101\rangle$. This means that a system can only ever have 2^n possible states each with its own probability of measurement.

To represent a six-sided die, the algorithm would have to ensure only 6 states were possible with a uniform distribution of the probabilities of each state. However, as explained earlier, a system can only ever have 2^n states. Initially, we sought to find a way to map a set of $[1,2^n]$ to a multiple of 6 such that $2^n = 6k, k \in \mathbb{Z}$. That way, if the n qubits are all placed into a state of superposition, over time, every possible state of the n-qubit system will have been measured with a uniform distribution. This would allow us to map the states, and by extension their distribution, to the set $[1,6]$ by binning them every k states. However, the prime factors of 6 are 2 and 3, and because of the factor of 3, the domain of 2^n cannot contain any multiples of 6. This makes it virtually impossible to simulate an ideal 6-sided die through quantum computing as it is now.

There are, however, some ways to come close to simulating dice. Two methods were investigated. The first method uses the lowest number of qubits n that contains the set $[1,6]$ in the set $[1,2^n]$. This is when $n = 3$. Thus, putting all 3 qubits into a state of superposition, the measured output of that system over time will produce a uniform distribution but for the numbers in the set [0,7]. The two extra states cannot be removed quantumly thus the probability of the die throws while random, will not be evenly distributed between 1 and 6, each with a probability of $0.1\dot{6}$, all the states would instead have a probability of 0.125. This comes close and if paired with a classical computer to simply ignore the two extra outputs, it could potentially work suitably.

The algorithm was to model one die. What happens when it is used to model the throwing of 2 dice simultaneously? This model was a slight compromise since there were only two extra states outside of the six that we wanted. In a 2 dice system we have a total 36 states, not all of which are unique. When this model is used to roll a dice twice, the total number of states that the quantum computer can generate now becomes 64. Before, the distribution was close and only 25% of the states were unusable. Now after the second dice roll, the overall shape of the distribution would be correct, however, now more than 40% of the states are unusable. Not only does this make the model harder to reduce to a set of $[2,12]$, but because of the shape of the distribution, depending on the initial choice for which states map to $[1,6]$, the distribution of the sum of the two dice rolls could end up significantly off.

The second method was adapted from Ed Swarts (2023). This method uses 5 qubits, each qubit is placed in a state of superposition and then measured. If R is number of "1's" counted in the measured output state, then the assigned dice roll would be equal to $R + 1$. An example relationship is outlined in Table 1 in Appendix A. This method produces a uniform distribution of every 5-bit bit pattern of which there are 32. The issue is that the number of "1's" are not distributed evenly among the bit patterns. This ultimately produces a distribution of the assigned rolls that is depicted in figure 5. This ultimately fails at properly reproducing an ideal six-sided die.

However, the distribution of the rolls is somewhat similar to the triangular distribution of the sums of two dice rolls. It failed to produce a decent enough model of a single 6-sided die, but it could potentially be used to make a model of 2 dice throws. In a 2 dice system, the sum of the rolls are contained in the

set [2,12], thus there are only 11 final states as shown in fig 1. Using 10 qubits and applying the same algorithm previously described, except now the assigned roll is given by R+2. This produces results that appear to follow a gaussian distribution, shown in figure 6. This model is the closest way we were able to simulate the two dice system completely natively.

It has been shown that there is currently no real way to perfectly simulate a 6-sided dice quantumly. For a 2 dice system, the complexity of the quantum model removes the benefits of the "randomness" that we were looking to take advantage of. This is because with the two methods listed in this report, the minimum number of qubits needed to represent the rolling of 2 dice simultaneously is 6. Recall that a quantum circuit with n qubits will have 2^n possible states. Thus, just to model a two dice system that throws both dice simultaneously, there are a minimum of 64 states that will be computed. This becomes even worse when looking at the second solution, since that requires 10 qubits, that comes out to 1024 states that need to be computed. This shows that at the current moment, it is not ideal to use a quantum computing approach to simulating non 2^n sided dice. The classical solutions well outperform the quantum solutions in terms of both simplicity and a much closer match to its statistical properties.

4. Conclusion

It has been shown that there is currently no real way to perfectly simulate a 6-sided dice quantumly. For a 2 dice system, the complexity of the quantum model removes the benefits of the "randomness" that we were looking to take advantage of. This is because with the two methods listed in this report, the minimum number of qubits needed to represent the rolling of 2 dice simultaneously is 6. Recall that a quantum circuit with qubits will have possible states. Thus, just to model a two dice system that throws both dice simultaneously, there are a minimum of 64 states that will be computed. This becomes even worse when looking at the second solution, since that requires 10 qubits, that comes out to 1024 states that need to be computed. This shows that at the current moment, it is not ideal to use a quantum computing approach to simulating non sided dice. The classical solutions well outperform the quantum solutions in terms of both simplicity and a much closer match to its statistical properties.

References

1. SYCUAN (2020) *HOW TO PLAY CASINO CRAPS FOR BEGINNERS, SYCUAN Casino Resort*. SYCUAN. Available at: https://www.sycuan.com/blog/how-to-play-craps-for-beginners/#:~:text=The%20concept%20of%20playing%20craps,"point"%20of%20the%20game. (Accessed: March 20, 2023).
2. Python Software Foundation. (2021). random — Generate pseudo-random numbers — Python 3.10.0 documentation. Available at: https://docs.python.org/3/library/random.html (Accessed: March 22, 2023).
3. Rohatgi, P. (2015). The randomness of randomness: True random number generators. IEEE Potentials, 34(2), 17-20. doi: 10.1109/MPOT.2014.2369165 (Accessed: March 24, 2023)
4. Swartz, E. (2022) *Create a quantum circuit to simulate rolling single die, Medium*. Medium. Available at: https://medium.com/@ed.swartz/create-a-quantum-circuit-to-simulate-rolling-single-die-e53dd44dd347 (Accessed: March 24, 2023).

Appendix A

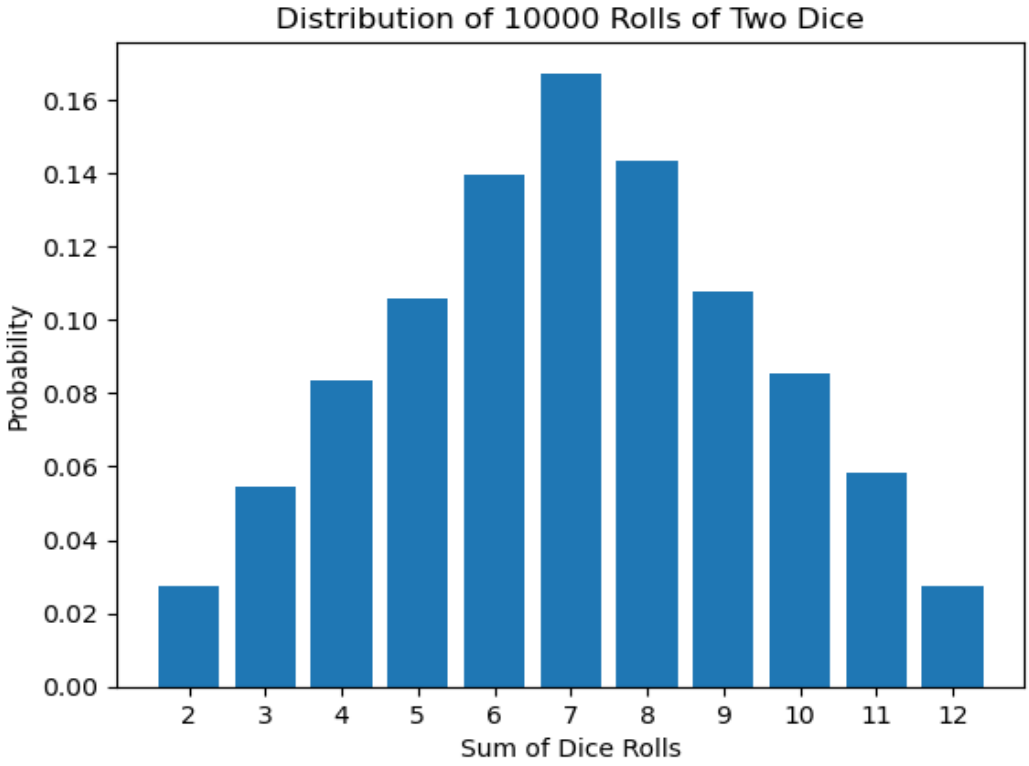

Figure 1: Distribution of 10000 Rolls of Two Dice

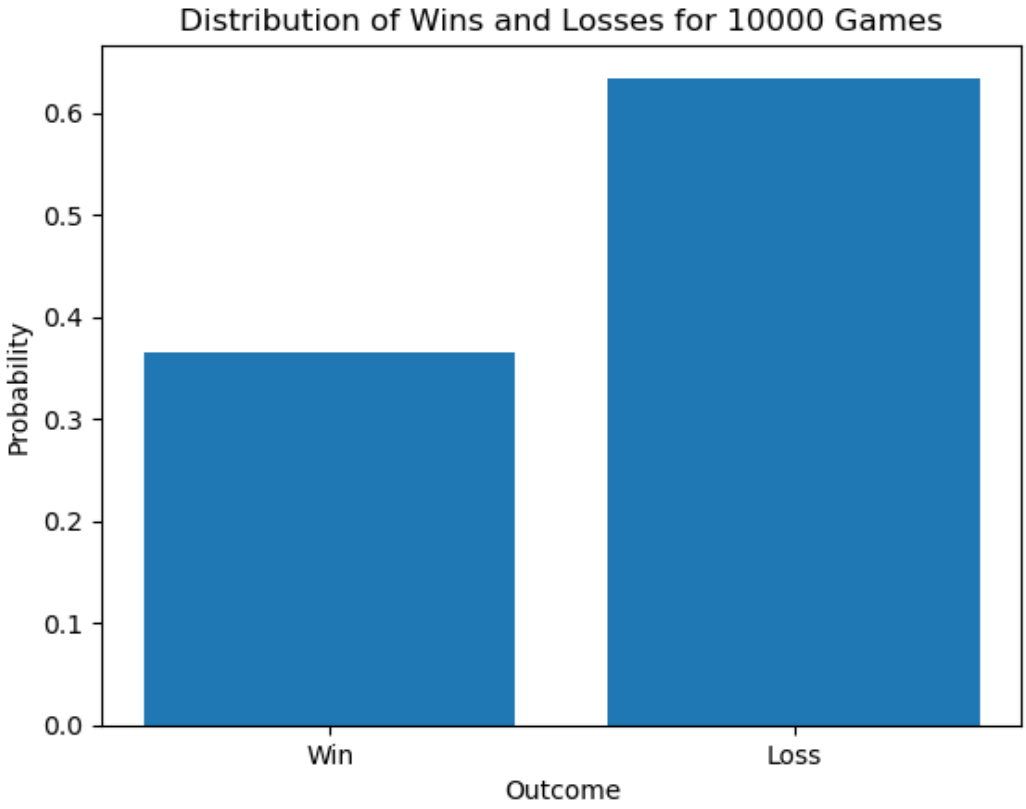

Figure 2: Distribution of Wins and Losses for 10000 Games

Table 1: Examples for Method 2

Bit Pattern	Count of 1's ("*R*")	Dice Value
00000	0	1
00001	1	2
00011	2	3
00111	3	4
01111	4	5
11111	5	6

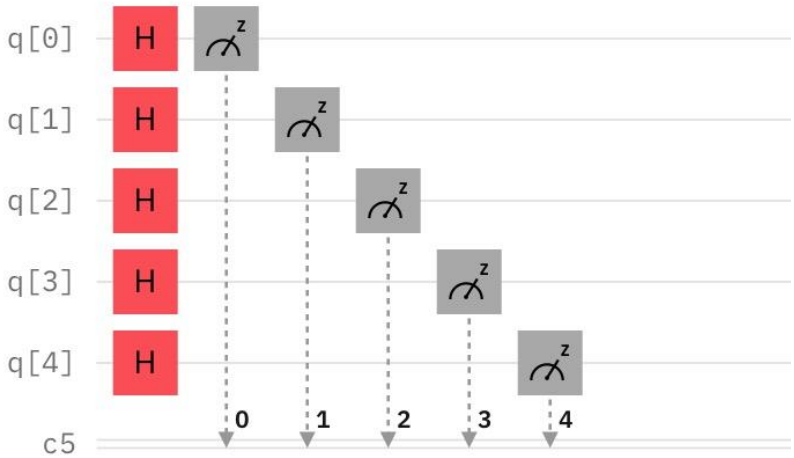

Figure 3: Quantum Circuit describing the second method for 1 die.

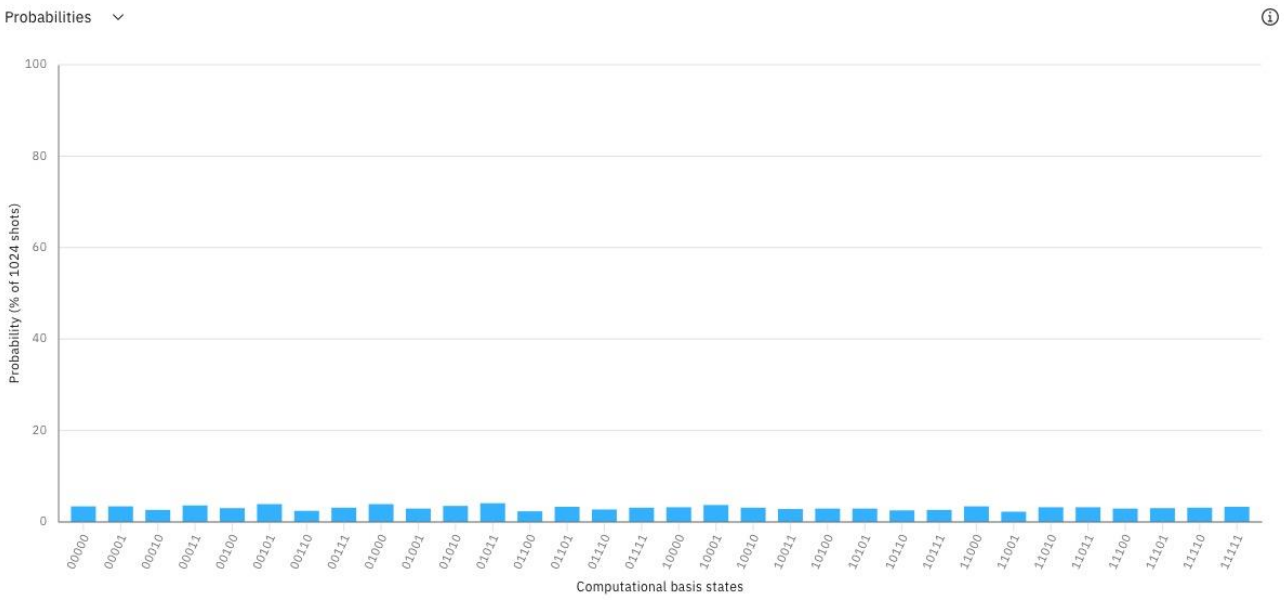

Figure 4: Probability distribution of 1024 rolls

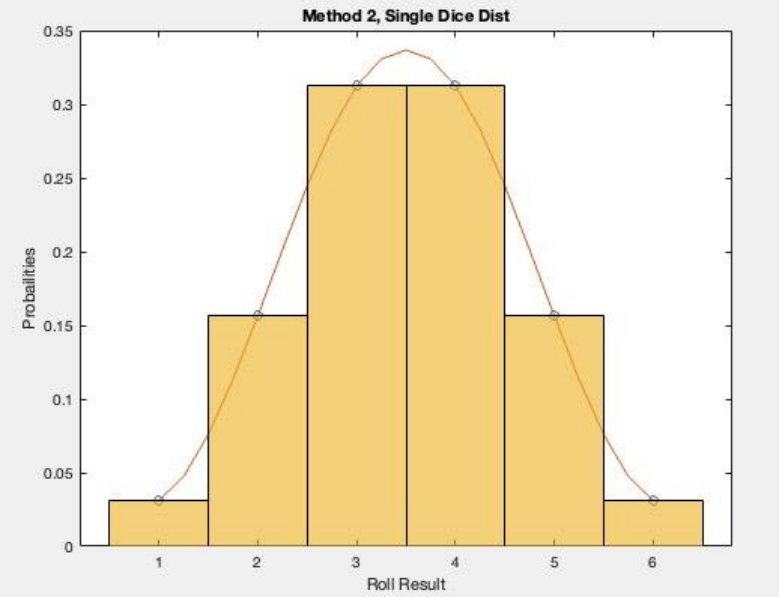

Figure 5: Distribution of Method 2 with a single die

Figure 6: Distribution of Method 2 with two dice rolls.