Analysis of Quantum Game Theoretic Models with a Python simulator

Indranil Ghosh
Jadavpur University, PG I, Physics Department
Email: indranilg49@gmail.com

December 20, 2018

Abstract

This project is to analyse and simulate the quantum versions of two game theoretic models: "Spin Flip Game" and "Prisoners dilemma", an interdisciplinary study of Quantum computation and Game theory. The applications of Quantum computation in Game theory have gone through a rapid development and have attracted a lot of interdisciplinary attentions. A Python program has been designed (https://github.com/indrag49/Quantum-SimuPy), to simulate the analysis of these quantum games. Packages used: numpy, pandas, random and matplotlib. Our objective is to use widely available python packages to design a simple quantum simulator that analyses simple game theory models.

1 Introduction

Quantum game theory is an interdisciplinary field of study that lies in the intersection of Game theory and Quantum Mechanics. This models the interactions of agents utilising quantum moves and has applications in quantum mechanics, economic game theory and population biology. In quantum games, the players perform local unitary operations on their qubits that correspond to their classical moves. Quantum games exhibit completely new properties that solely differ from that of the corresponding classical games.

2 "Spin Flip Game"

Let this game be played between two players: Alice and Bob and we consider Alice to always play the first move. The rules set are very simple. Out of two possible electron spin states 'up' and 'down', Alice sets any one to begin with, according to her choice. But let's say, Alice is 'supposed' to set the initial spin to be in the 'up' state. Without knowing what Alice has set, Bob plays one of σ_x and I operators. Again Alice, not knowing Bob's move and the present state of the electron also applies σ_x or I. At last Bob plays again either σ_x or I and the final state is measured. If the final spin is in the 'up' state Alice loses 1 point and Bob gains 1 point and if it is in the 'down' state Alice gains 1 point and Bob loses 1 point. We later introduce some complicated situations where Alice or Bob or both cheat.

Here, 'up' spin is defined as,

$$u = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{1}$$

And the 'down' spin as,

$$d = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2}$$

2.1 Game representation and the Payoff Matrix

The above mentioned game is a two-person, zero-sum game, as the payoffs of Bob and Alice are exactly opposite to each other. The game can be represented by the binary tree given below:

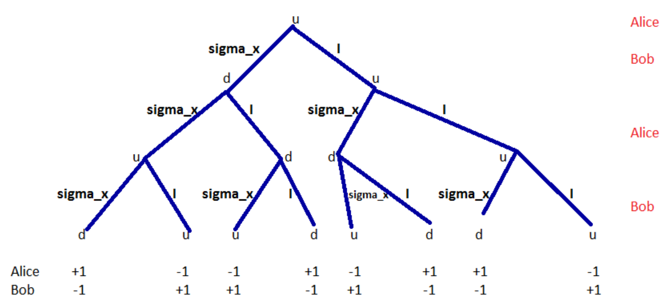

Figure 1: The basic spin flip game

The Payoff matrix of the game is represented as:

Alice/ Bob	I, I	I, σ_x	σ_x, I	σ_x, σ_x
I	-1, 1	1, -1	1, -1	-1, 1
σ_x	1, -1	-1, 1	-1, 1	1, -1

In this game, there will be mixed strategies for both Alice and Bob. Alice chooses moves σ_x with probability $\frac{1}{2}$ and I with probability $\frac{1}{2}$. Same is the case for Bob. Looking at the payoff matrix above, we see that the expected payoff $\bar{\pi_A}$ of Alice, despite of what Bob's move is, is $\bar{\pi_A} = \frac{1}{2}(1) + \frac{1}{2}(-1) = 0$. Similarly, Bob's expected payoff $\bar{\pi_B}$ is, $\bar{\pi_B} = \frac{1}{4}(1) + \frac{1}{4}(-1) + \frac{1}{4}(1) + \frac{1}{4}(-1) = 0$. If we consider 'N' games and 'x' wins for Alice, the actual payoff to Alice will be a member of the payoff set, $\Pi = \{f(x;N)\} = \{2x-N, x=0,1,...,N\}$ and the probability of these payoffs,$P(\Pi) = \{f(x;N,p)\} = \{^{N}C_{x}p^{x}q^{N-x}, for x=0,1,...,N\}$. Same can be computed for Bob.

Here, we have taken N=1, so, $\Pi = \{-1,+1\}$ and $P(\Pi) = \{\frac{1}{2},\frac{1}{2}\}$. Now we use the software to simulate the basic spin flip game.

```
def spin_flip1():
    #Alice prepares the initial state
    u=Q0
    d=Q1
    A=u
    state=A

    #Bob plays his move: either sigma_x or I2
    r=random.uniform(0, 1)
    B=PauliX(state) if r<0.5 else I2.dot(state)
    state=B

    #Now again Alice plays
    r=random.uniform(0, 1)
    A=PauliX(state) if r<0.5 else I2.dot(state)
    state=A

    #Last move by Bob
    r=random.uniform(0, 1)
    B=PauliX(state) if r<0.5 else I2.dot(state)
    state=B

    plot_measure(measure(state))

spin_flip1()
```

The final states:

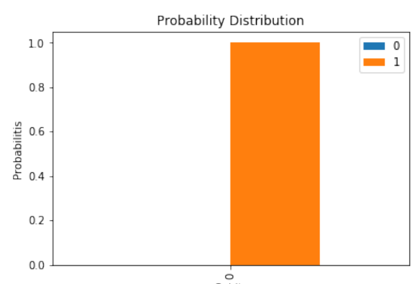

Figure 2

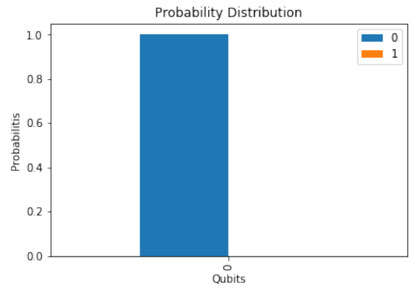

Figure 3: measurement of final state

On measuring the final state we see that there is an equal probability for both 'up' and 'down' spins to be the final state. The probability of winning for Alice is same as for Bob. Next we consider the cases where one of them cheats or both of them cheat.

2.2 Alice cheats

Now, we know Alice is always supposed to set the initial spin in the 'up' state. Let's say, Alice, instead of setting it in the 'up' state sets it in the 'down' state to initialize with, without Bob's knowledge. The successive moves by the players are same as before. What happens is, the arrangements of payoffs for both Alice and Bob reverses, i.e, +1 becomes -1 and vice-versa. Thus the set of payoffs, Π and the corresponding payoff probabilities $P(\Pi)$ for both Alice and Bob remain unchanged. So Alice's cheating goes in vain.

Now, Alice tries to be more clever and instead of just inversing the initial spin, she sets the initial state as a superposition of both 'up' and 'down' spins. Let this state be $\frac{1}{\sqrt{2}}[u+d]$.
We see that

$$I[\frac{1}{\sqrt{2}}(u+d)] = \frac{1}{\sqrt{2}}(Iu+Id) = \frac{1}{\sqrt{2}}(u+d) \tag{3}$$

and

$$\sigma_x[\frac{1}{\sqrt{2}}(u+d)] = \frac{1}{\sqrt{2}}(\sigma_x u+\sigma_x d) = \frac{1}{\sqrt{2}}(d+u) = \frac{1}{\sqrt{2}}(u+d) \tag{4}$$

The binary tree that represents the game:

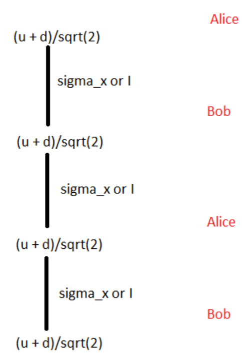

Figure 4: Game Tree when only Alice cheats

Using the software:

```
>>> u=Q0
>>> d=Q1
>>> A_initial=(u+d)/np.sqrt(2)
>>> A_initial
array([0.70710678, 0.70710678])
>>> I2.dot(A_initial)
array([0.70710678, 0.70710678])
>>> PauliX(A_initial)
array([0.70710678, 0.70710678])
```

We simulate the game:

```
def spin_flip2():
    #Alice prepares the initial state
    u=Q0
    d=Q1
    A=(u+d)/np.sqrt(2)
    state=A

    #Bob plays his move: either sigma_x or I2
    r=random.uniform(0, 1)
    B=PauliX(state) if r<0.5 else I2.dot(state)
    state=B

    #Now again Alice plays
    r=random.uniform(0, 1)
    A=PauliX(state) if r<0.5 else I2.dot(state)
    state=A
```

```
    #Last move by Bob
    r=random.uniform(0, 1)
    B=PauliX(state) if r<0.5 else I2.dot(state)
    state=B

    plot_measure(measure(state))


spin_flip2()
```

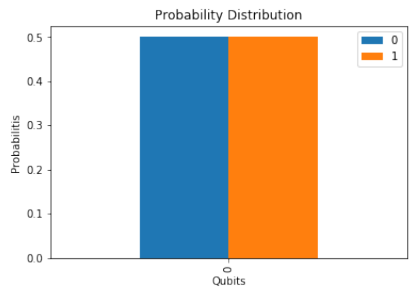

Figure 5: measurement of final state

Alice discovers on measurement of the final state that, once again she looses or gains 1 point with equal probability. For N=1, again, $\Pi = \{-1, 1\}$ and $P(\Pi) = \{(\frac{1}{\sqrt{2}})^2, (\frac{1}{\sqrt{2}})^2\} = \{\frac{1}{2}, \frac{1}{2}\}$ for both Alice and Bob.

$$|\Psi\rangle = \frac{|u\rangle + |d\rangle}{\sqrt{2}} \tag{5}$$

Measurement yields base state $|u\rangle$ with probability, $|\langle u|\Psi\rangle|^2 = (\frac{1}{\sqrt{2}})^2 = \frac{1}{2}$ and base state $|d\rangle$ with probability $|\langle d|\Psi\rangle|^2 = (\frac{1}{\sqrt{2}})^2 = \frac{1}{2}$. We should keep in mind that u and d are always orthonormal.

2.3 Bob cheats

Now, let's say, Alice remains innocent and plays fair, but Bob commits a fraud in his move. Instead of applying σ_x or I, he plays the Hadamard operation: $H = \frac{(\sigma_x + \sigma_z)}{\sqrt{2}}$.

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{6}$$

After Bob's 1st move, the state of the spin will be in s superposed state:

$$Hu = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}(u + d) \tag{7}$$

We have noticed in the previous section that the application of σ_x or I on $\frac{(u+d)}{\sqrt{2}}$ does not change its state. So Bob's second move, i.e, application of Hadamard operator again brings back state 'up' again.

$$H(Hu) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = u \tag{8}$$

So, Bob always wins. The game can be represented by a binary tree:

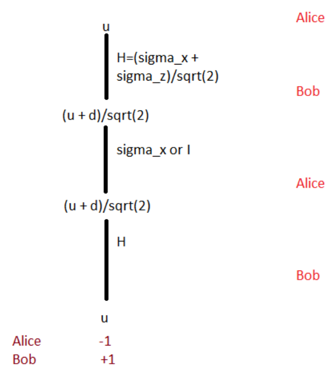

Figure 6: Game Tree when only Bob cheats

Using the software:

```
>>> u=Q0
>>> d=Q1
>>> A=u
>>> Hadamard(A)
array([0.70710678, 0.70710678])
>>> PauliX(Hadamard(A))
array([0.70710678, 0.70710678])
>>> I2.dot(Hadamard(A))
array([0.70710678, 0.70710678])
>>> Hadamard(PauliX(Hadamard(A)))
array([1., 0.])
```

We simulate the game:

```
def spin_flip3():
    #Alice prepares the initial state
    u=Q0
    d=Q1
    A=u
```

```
    state=A

    #Bob plays his move: Hadamard operator
    B=Hadamard(state)
    state=B

    #Now again Alice plays
    r=random.uniform(0, 1)
    A=PauliX(state) if r<0.5 else I2.dot(state)
    state=A

    #Last move by Bob: again Hadamard operator
    B=Hadamard(state)
    state=B

    plot_measure(measure(state))

spin_flip3()
```

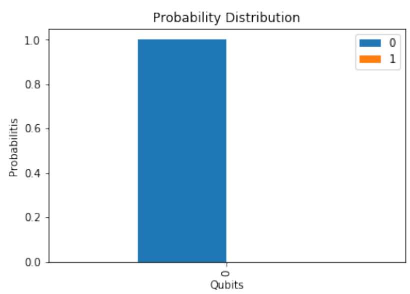

Figure 7: measurement of final state

The payoff set Π for both Alice and Bob is $\{-1, 1\}$. The final state, $|\Psi\rangle = |u\rangle$. Measurement yields base state 'u' with probability, $|\langle u|\Psi\rangle|^2 = 1$ and base state 'd' with probability $|\langle d|\Psi\rangle|^2 = 0$. So, the corresponding probabilities $P(\Pi)$ for Bob is $\{0, 1\}$ and for Alice is $\{1, 0\}$.

2.4 Both Alice and Bob cheat

Now, let's consider a situation where both Alice and Bob cheats. There can be two cases where Alice sets the initial spin to be in 'down' state or the superposed state stated above, i.e, $\frac{u+d}{\sqrt{2}}$. Alice then plays σ_x or I and Bob always plays the Hadamard operator H, in the successive game steps.

2.4.1 Case 1

Let's suppose Alice sets the initial spin to be in 'down' state. Bob plays H,

$$Hd = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}0\\1\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\-1\end{bmatrix} = \frac{1}{\sqrt{2}}(u-d) \tag{9}$$

Now Alice plays σ_x or I

$$\sigma_x(Hd) = \frac{1}{\sqrt{2}}\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}1\\-1\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}-1\\1\end{bmatrix} = \frac{1}{\sqrt{2}}(d-u) \tag{10}$$

and

$$I(Hd) = \frac{1}{\sqrt{2}}(u-d) \tag{11}$$

After Alice's move, Bob again plays the H operator,

$$H(\sigma_x(Hd)) = \frac{1}{2}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}-1\\1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}0\\-2\end{bmatrix} = \begin{bmatrix}0\\-1\end{bmatrix} = -d \tag{12}$$

$$H(I(Hd)) = \frac{1}{2}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}1\\-1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}0\\2\end{bmatrix} = \begin{bmatrix}0\\1\end{bmatrix} = d \tag{13}$$

The game can be represented by the binary tree:

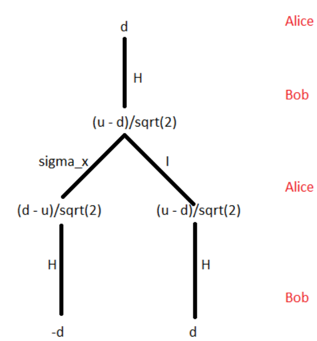

Figure 8: Game Tree when both Alice and Bob cheat (case 1)

The payoff set Π for both Alice and Bob is {-1, 1}. When the final state $|\Psi\rangle = |-d\rangle$, measurement yields base state 'down' with probability $|\langle d|\Psi\rangle|^2 = |\langle d| - d\rangle|^2 = |-1|^2 = 1$ and the base state 'up' with probability $|\langle u|\Psi\rangle|^2 = |\langle u| - d\rangle|^2 = 0$. So the corresponding probability set for the payoff set for Alice is $P(\Pi) = \{0, 1\}$ and for Bob is $P(\Pi) = \{1, 0\}$. So, Alice always wins.
Using the software:

```
>>> u=Q0
>>> d=Q1
>>> A=d
>>> Hadamard(A)
array([ 0.70710678, -0.70710678])
>>> PauliX(Hadamard(A))
array([- 0.70710678, 0.70710678])
>>> I2.dot(Hadamard(A))
array([ 0.70710678, -0.70710678])
>>> Hadamard(PauliX(Hadamard(A)))
array([ 0., -1.])
>>> Hadamard(I2.dot(Hadamard(A)))
array([ 0., 1.])
```

We simulate the game:

```
def spin_flip4():
    #Alice prepares the initial state
    u=Q0
    d=Q1
    A=d
    state=A

    #Bob plays his move: Hadamard operator
    B=Hadamard(state)
    state=B

    #Now again Alice plays
    r=random.uniform(0, 1)
    A=PauliX(state) if r<0.5 else I2.dot(state)
    state=A

    #Last move by Bob: again Hadamard operator
    B=Hadamard(state)
    state=B

    plot_measure(measure(state))

spin_flip4()
```

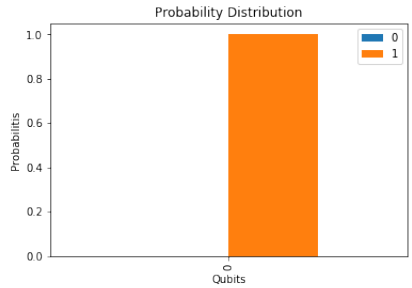

Figure 9: measurement of final state

2.4.2 Case 2

Let's suppose Alice sets the initial spin to be in a superposed state, $\frac{u+d}{\sqrt{2}}$. Bob plays H

$$H(\frac{u+d}{\sqrt{2}}) = u \tag{14}$$

Now Alice plays either σ_x or I

$$\sigma_x(u) = d \tag{15}$$

$$I(u) = u \tag{16}$$

Bob at last again plays H,

$$H(d) = \frac{1}{\sqrt{2}}(u - d) \tag{17}$$

$$H(u) = \frac{1}{\sqrt{2}}(u + d) \tag{18}$$

The game can be represented by the binary tree:

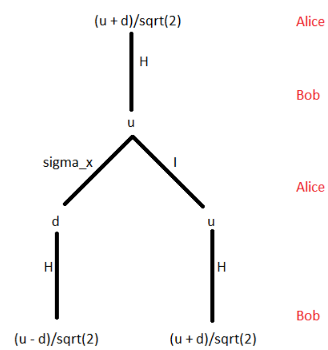

Figure 10: Game Tree when both Alice and Bob cheat (case 2)

The payoff set Π for both Alice and Bob is -1, 1. When $|\Psi\rangle = \frac{|u\rangle - |d\rangle}{\sqrt{2}}$, measurement yields base state 'up' with probability, $|\langle u|\Psi\rangle|^2 = |\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$ and base state 'down' with probability, $|\langle d|\Psi\rangle|^2 = |-\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$.When $|\Psi\rangle = \frac{|u\rangle + |d\rangle}{\sqrt{2}}$, measurement yields base state 'up' with probability, $|\langle u|\Psi\rangle|^2 = |\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$ and base state 'down' with probability, $|\langle d|\Psi\rangle|^2 = |\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$. The corresponding probability for payoff set for both ALice and Bob is $\{\frac{1}{2}, \frac{1}{2}\}$. So, there is an equal probability of winning or losing for both Alice and Bob.
Using the software:

```
def spin_flip5():
    #Alice prepares the initial state
    u=Q0
    d=Q1
    A=(u+d)/np.sqrt(2)
    state=A

    #Bob plays his move: Hadamard operator
    B=Hadamard(state)
    state=B

    #Now again Alice plays
    r=random.uniform(0, 1)
    A=PauliX(state) if r<0.5 else I2.dot(state)
    state=A

    #Last move by Bob: again Hadamard operator
    B=Hadamard(state)
    state=B
```

```
    plot_measure(measure(state))

spin_flip5()
```

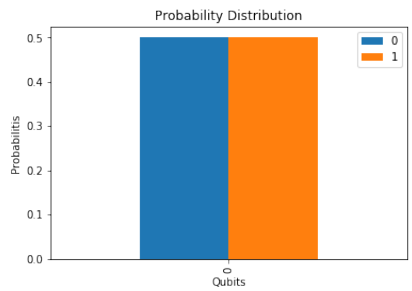

Figure 11: measurement of final state

3 Quantum Prisoner's Dilemma

Let us discuss the classical version of the Prisoner's dilemma at first. It is a simple 2 X 2 game that is not zero sum and is usually presented in static form. Suppose there are two suspects, Alice and Bob, suspected of committing a crime together, are brought in for interrogation. Each has two possible moves, "cooperate" (C), i.e, remains silent or "defect" (D), i.e, confesses for the crime. The game can be represented by a binary tree:

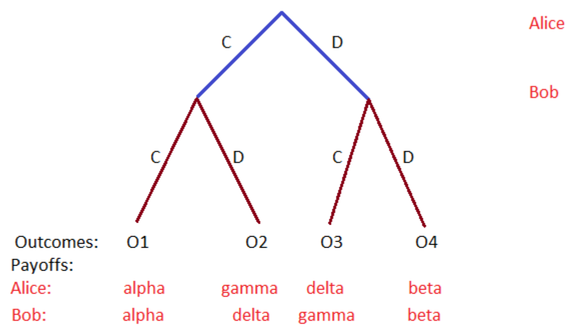

Figure 12: Prisoner's Dilemma

And the corresponding payoff matrix:

Alice/ Bob	C	D
C	(α, α)	(γ, δ)
D	(δ, γ)	(β, β)

Here, α, β, γ and δ are the payoffs satisfying the following chain of inequalities: $\delta > \alpha > \beta > \gamma$. Let's assume the game is symmetric, i.e., the payoffs α, β, γ and δ are the same for each player and also that the payoffs have ordinal significance. Here, strategies, S_A={C, D} and S_B={C, D} and payoffs, $\Pi = \{\pi_A, \pi_B\}$.

Now, let's dig deeper into the game tree. We at first consider the case of Alice cooperating. Now, if Bob cooperates too, he gets an α and if he defects, he gets a δ. As $\delta > \alpha$, Bob is better off defecting. Now, if Alice defects, Bob gets a γ for cooperating and a β for defecting. Again $\beta > \gamma$ and Bob is better of defecting. So, the dominant strategy for Bob is the move s_B, such that the payoff π_B to Bob has the property,

$$\pi_B(s_A^i, s_B) \geq \pi_B(s_A^i, s_B^j) \tag{19}$$

for $\forall$ $s_A^i \in S_A$ and $s_B^j \in S_B$. So in our first case, the dominant strategy for Bob is to defect. $s_B = D$. Similarly it can be shown that $s_A = D$. Thus, two rational players will defect and receive a payoff of β each, while two irrational players can cooperate and receive a greater payoff of α each. The game will be in equilibrium with $\{s_A, s_B\} = \{D, D\}$ and $\{\pi(s_A), \pi(s_B)\} = \{\beta, \beta\}$. This outcome is referred to as the "Prisoner's Dilemma" because both Alice and Bob would be better off if both of them cooperated, yielding $\pi_A = \pi_B = \alpha$.

When a player has a strictly dominant strategy, it is irrational for that player to choose any other strategy, as he is guaranteed a lower payoff. Thus in this game, individual rationality leads to the dilemma despite the fact that both players would be better off by cooperating. Next we use the **IDSDS**(*The Iterated Deletion of Strictly Dominated Strategies*) algorithm. The output of this algorithm if contains a single strategy profile is called the *iterated strictly dominant-strategy equilibrium.* It can be shown that if there is a common knowledge of rationality, the only strategy profile surviving the IDSDS algorithm can be played.

The iteration of IDSDS:

1)

Alice/ Bob	C	D
C	(α, α)	(γ, δ)
D	(δ, γ)	(β, β)

2) Alice's C is strictly dominated by D

Alice/ Bob	C	D
D	(δ, γ)	(β, β)

3) Bob's C is strictly dominated by D

Alice/ Bob	D
D	(β, β)

We see that {D, D} is the *iterated strictly dominant-strategy equilibrium.* This is the only strategy profile that can be played, if there is a common knowledge of rationality. For our covinience let's consider $\alpha = 3$, $\beta = 1$, $\gamma = 0$ and $\delta = 5$. So,

Alice/ Bob	C	D
C	(3, 3)	(0, 5)
D	(5, 0)	(1, 1)

```
alpha, beta, gamma, delta=3, 1, 0, 5
Alice=np.array(([alpha, gamma], [delta, beta]))
Bob=np.array(([alpha, delta], [gamma, beta]))

def IDSDS(P1, P2):
    check1, check2=0, 0
    while True:
```

```
        for i in range(len(P1)):
            for j in range(len(P1)):
                if list(P1[i, :]<P1[j, :]).count(True)==len(P1[0]):
                    P1=np.delete(P1, (i), axis=0)
                    P2=np.delete(P2, (i), axis=0)
                    check1=1
                    break
            if check1==1: break

        if check1==0:
            for i in range(len(P2[0])):
                for j in range(len(P2[0])):
                    if list(P2[:, i]<P2[:, j]).count(True)==len(P2):
                        P2=np.delete(P2, (i), axis=1)
                        P1=np.delete(P1, (i), axis=1)
                        check2=1
                        break
                if check2==1:break

        print(P1)
        print(P2)
        if check1==0 and check2==0: break
        check1, check2=0, 0

    return (P1, P2)

IDSDS(Alice, Bob)
```

We see that the output of the program matches the theory. We must also notice that {D, D}, yielding payoffs $\{\beta, \beta\}$ is the only strong *Nash equilibrium* of the game. It is because neither Alice nor Bob can increase his/her payoff by unilaterally departing from the given equilibrium point. We run a python program:

```
alpha, beta, gamma, delta=3, 1, 0, 5
Alice=np.array(([alpha, gamma], [delta, beta]))
Bob=np.array(([alpha, delta], [gamma, beta]))

L1=[]
L2=[]
l=len(Alice)
for i in range(l):
    a=Alice[:, i]
    M=max(a)
    for j in range(l):
        if a[j]==M: L1+=[[j, i], ]
```

```
for i in range(l):
    a=Bob[i, :]
    M=max(a)
    for j in range(l):
        if a[j]==M: L2+=[[i, j], ]

N=[i for i in L1 if i in L2]

I1={0:'C', 1:'D'}
I2={0:'C', 1:'D'}

Nash=[(I1[i], I2[j]) for [i, j] in N]
## The Nash equilibria
Nash
```

and notice that we get [('D', 'D')] as the output. Also it must be pointed out that the point $\{\alpha, \alpha\}$ is *Pareto optimal.* Now that we have had a clear introduction to the classical version, in the next subsection, the quantum version of the Prisoner's dilemma game is introduced and is shown how addition of quantum moves changes the outcome of the game and how the dilemma can be escaped.

3.1 Quantum version of the Prisoner's Dilemma

The quantum game Γ, defined as an interaction between two or more players, consists the elements: Γ=$\Gamma(\mathbf{H}, \Lambda, \{s_i\}_j, \{\pi_i\}_j)$. H is a Hilbert space, Λ is the initial prepared state of the game, $\{s_i\}_j\}$ is the set of moves of player j, while $\{\pi_i\}_j$ is the set of payoffs to player j. The objectives of the quantum version of the game are to maximize the payoffs to player j and also find a way to escape the dilemma.

Alice and Bob both possess a qubit each and are allowed to manipulate his/her own qubit. Eqch qubit lies in $\mathbf{H}_2$ which as basis vectors $|C\rangle$ and $|D\rangle$. The game lies in $\mathbf{H}_2 \otimes \mathbf{H}_2$ with basis vectors $|CC\rangle$, $|CD\rangle$, $|DC\rangle$ and $|DD\rangle$. Following the convention, the left most qubit belongs to Alice and the right most qubit to Bob.

Let's map $|C\rangle \rightarrow |0\rangle$ and $|D\rangle \rightarrow |1\rangle$. The initial state of the game,

$$|\Lambda\rangle = U\,|00\rangle \tag{20}$$

U is a unitary operator, known both to Alice and Bob, that operates on both the qubits. The purpose of U is to entangle Alice's and Bob's qubits and without U the payoffs to Alice and Bob remains same as that of the classical game. The strategic moves are s_A and s_B belonging to Alice and Bob respectively. Here, $s_A(= U_A)$ and $s_B(= U_B)$ are unitary matrices that operate on the respective player's qubit only. After the moves played by both Alice and Bob, the state of the game is,

$$|\Psi_s\rangle = (U_A \otimes U_B)\Lambda = (U_A \otimes U_B)U\,|00\rangle \tag{21}$$

For the final state of the game, we need to find the adjoint of the matrix U. So, $U^\dagger = (U^*)^T$. Now, U is unitary. So, $U^\dagger = U^{-1}$. The final state of the game is,

$$|\Psi_f\rangle = U^\dagger\,|\Psi_s\rangle = U^\dagger(U_A \otimes U_B)U\,|00\rangle \tag{22}$$

On measurement of the final state of the system, we get, the expected payoff to Alice,

$$\overline{\pi_A} = \alpha|\,\langle\Psi_f|00\rangle\,|^2 + \gamma|\,\langle\Psi_f|01\rangle\,|^2 + \delta|\,\langle\Psi_f|10\rangle\,|^2 + \beta|\,\langle\Psi_f|11\rangle\,|^2 \tag{23}$$

and the expected payoff to Bob,

$$\overline{\pi_B} = \alpha|\,\langle\Psi_f|00\rangle\,|^2 + \delta|\,\langle\Psi_f|01\rangle\,|^2 + \gamma|\,\langle\Psi_f|10\rangle\,|^2 + \beta|\,\langle\Psi_f|11\rangle\,|^2 \tag{24}$$

We consider

$$U = \frac{(I \otimes I + i\sigma_x \otimes \sigma_x)}{\sqrt{2}} \tag{25}$$

So, the inverse

$$U^{-1} = \frac{(I \otimes I - i\sigma_x \otimes \sigma_x)}{\sqrt{2}} \tag{26}$$

We know, $U^\dagger = U^{-1}$. Now to prepare the initial state,

$$|\Lambda\rangle = U\,|00\rangle = \frac{(|00\rangle + i\,|11\rangle)}{\sqrt{2}} \tag{27}$$

After the initial state is prepared, we set some moves by Alice and Bob. For cooperating, we apply I and for defecting we apply σ_x.

Using the software, we define the game QPD():

```
def QPD(U_A, U_B, alpha, beta, gamma, delta):
    sigma_x=PauliX(I2)
    U=(np.kron(I2, I2)+1j*np.kron(sigma_x, sigma_x))/np.sqrt(2)
    U_dag=np.conj(U.T)

    initial=U.dot(Q00)

    PsiS=np.kron(U_A, U_B).dot(initial)
    PsiF=U_dag.dot(PsiS)
    plot_measure(measure(PsiF))

    cpsif=np.conj(PsiF.T)

    def pi(alpha, beta, gamma, delta): return (alpha*np.abs(cpsif.dot(Q00))**2+
```

```
gamma*np.abs(cpsif.dot(Q01))**2+delta*np.abs(cpsif.dot(Q10))**2+
beta*np.abs(cpsif.dot(Q11))**2, alpha*np.abs(cpsif.dot(Q00))**2+
delta*np.abs(cpsif.dot(Q01))**2+gamma*np.abs(cpsif.dot(Q10))**2+
beta*np.abs(cpsif.dot(Q11))**2)

return pi(alpha, beta, gamma, delta)
```

3.1.1 Both cooperate

$U_A = U_B = I$. so,

$$|\Psi_s\rangle = (I \otimes I)\,|\Lambda\rangle = (I \otimes I)U\,|00\rangle = \frac{1}{\sqrt{2}}(I \otimes I)[|00\rangle + i\,|11\rangle] = \frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] \tag{28}$$

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger\,|\Psi_s\rangle && (29)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + i\,|11\rangle] && (30)\\
&= \frac{1}{2}[|00\rangle - i\,|11\rangle + i\,|11\rangle + |00\rangle] && (31)\\
&= \frac{1}{2}2\,|00\rangle && (32)\\
&= |00\rangle && (33)
\end{aligned}$$

We get $|00\rangle$ with probability 1. As in the classical case we consider $\alpha = 3$, $\beta = 1$, $\gamma = 0$, $\delta = 5$. So, $\overline{\pi_A} = 3$ and $\overline{\pi_B} = 3$.

```
QPD(I2, I2, 3, 1, 0, 5)
```

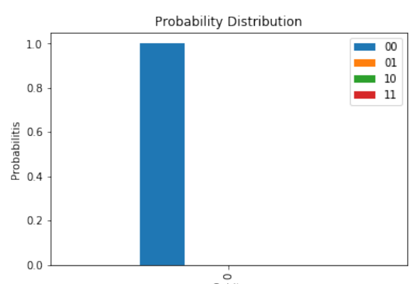

Figure 13: measurement of final state

```
>>> (2.9999999999999987, 2.9999999999999987)
```

3.1.2 Alice defects, Bob cooperates

$U_A = \sigma_x, U_B = I$. so,

$$\begin{aligned}
|\Psi_s\rangle &= (\sigma_x \otimes I)\,|\Lambda\rangle && (34)\\
&= (\sigma_x \otimes I)U\,|00\rangle && (35)\\
&= \frac{1}{\sqrt{2}}(\sigma_x \otimes I)[|00\rangle + i\,|11\rangle] && (36)\\
&= \frac{1}{\sqrt{2}}[|10\rangle + i\,|01\rangle] && (37)
\end{aligned}$$

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger\,|\Psi_s\rangle && (38)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|10\rangle + i\,|01\rangle] && (39)\\
&= \frac{1}{2}[|10\rangle - i\,|01\rangle + i\,|01\rangle + |10\rangle] && (40)\\
&= \frac{1}{2}2\,|10\rangle && (41)\\
&= |10\rangle && (42)
\end{aligned}$$

We get $|10\rangle$ with probability 1. $\overline{\pi_A} = 5$ and $\overline{\pi_B} = 0$.

```
sigma_x=PauliX(I2)
QPD(sigma_x, I2, 3, 1, 0, 5)
```

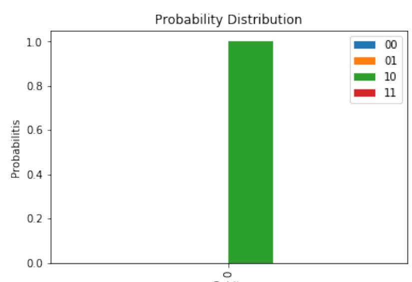

Figure 14: measurement of final state

```
>>> (4.999999999999998, 0.0)
```

3.1.3 Alice cooperates, Bob defects

$U_A = I, U_B = \sigma_x$. so,

$$|\Psi_s\rangle = (I \otimes \sigma_x) |\Lambda\rangle \tag{43}$$
$$= (I \otimes \sigma_x) U |00\rangle \tag{44}$$
$$= \frac{1}{\sqrt{2}} (I \otimes \sigma_x)[|00\rangle + i |11\rangle] \tag{45}$$
$$= \frac{1}{\sqrt{2}} [|01\rangle + i |10\rangle] \tag{46}$$

$$|\Psi_f\rangle = U^\dagger |\Psi_s\rangle \tag{47}$$
$$= \frac{1}{2} (I \otimes I - i\sigma_x \otimes \sigma_x)[|01\rangle + i |10\rangle] \tag{48}$$
$$= \frac{1}{2} [|01\rangle - i |10\rangle + i |10\rangle + |01\rangle] \tag{49}$$
$$= \frac{1}{2} 2 |01\rangle \tag{50}$$
$$= |01\rangle \tag{51}$$

We get $|10\rangle$ with probability 1. $\overline{\pi_A} = 0$ and $\overline{\pi_B} = 5$.

```
sigma_x=PauliX(I2)
QPD(I2, sigma_x, 3, 1, 0, 5)
```

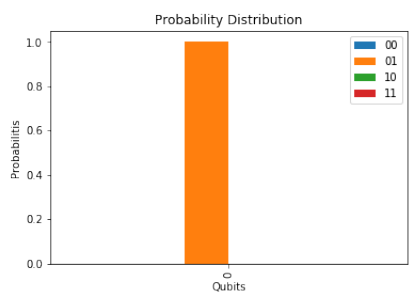

Figure 15: measurement of final state

```
>>> (0.0, 4.999999999999998)
```

3.1.4 Both defect

$U_A = \sigma_x, U_B = \sigma_x$. so,

$$\begin{aligned} |\Psi_s\rangle &= (\sigma_x \otimes \sigma_x)\,|\Lambda\rangle && (52)\\ &= \frac{1}{\sqrt{2}}[|11\rangle + i\,|00\rangle] && (53) \end{aligned}$$

$$\begin{aligned} |\Psi_f\rangle &= U^\dagger\,|\Psi_s\rangle && (54)\\ &= \frac{1}{2}(I\otimes I - i\sigma_x \otimes \sigma_x)[|11\rangle + i\,|00\rangle] && (55)\\ &= |11\rangle && (56) \end{aligned}$$

We get $|11\rangle$ with probability 1. $\overline{\pi_A} = 1$ and $\overline{\pi_B} = 1$.

```
sigma_x=PauliX(I2)
QPD(sigma_x, sigma_x, 3, 1, 0, 5)
```

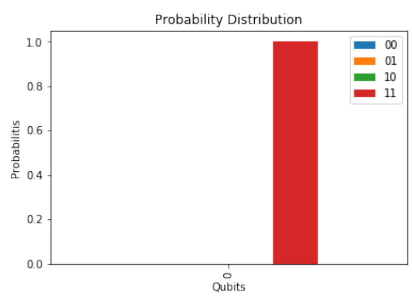

Figure 16: measurement of final state

```
>>> (0.9999999999999996, 0.9999999999999996)
```

3.1.5 Alice plays I and Bob plays H

We now consider some less traditional quantum moves by Alice and Bob. $U_A = I, U_B = H$. so,

$$\begin{aligned} |\Psi_s\rangle &= (I \otimes H)\,|\Lambda\rangle && (57)\\ &= \frac{1}{\sqrt{2}}[|0\rangle\,\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + i\,|1\rangle\,\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)] && (58)\\ &= \frac{1}{\sqrt{2}}[|00\rangle + |01\rangle + i\,|10\rangle - i\,|11\rangle] && (59) \end{aligned}$$

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle & (60)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + |01\rangle + i\,|10\rangle - i\,|11\rangle] & (61)\\
&= \frac{1}{2\sqrt{2}}[|00\rangle - i\,|11\rangle + |01\rangle - i\,|10\rangle + i\,|10\rangle + |01\rangle - i\,|11\rangle - |00\rangle] & (62)\\
&= \frac{1}{2\sqrt{2}}2[|01\rangle - i\,|11\rangle] & (63)\\
&= \frac{1}{\sqrt{2}}[|01\rangle - i\,|11\rangle] & (64)
\end{aligned}
$$

Here, $|\frac{1}{\sqrt{2}}|^2 = \frac{1}{2}$ and $|\frac{-i}{\sqrt{2}}|^2 = \frac{1}{2}$. So, a measurement of the final state yields an equal probability for a payout of $\gamma(0)$ or a payout of $\beta(1)$ to Alice and an equal probability for a payout of $\delta(5)$ or a payout of $\beta(1)$ to Bob (look at the payoff matrix to remove confusion). Here, $\overline{\pi_A} = 0.5$ and $\overline{\pi_B} = 3$.

```
H=Hadamard(I2)
QPD(I2, H, 3, 1, 0, 5)
```

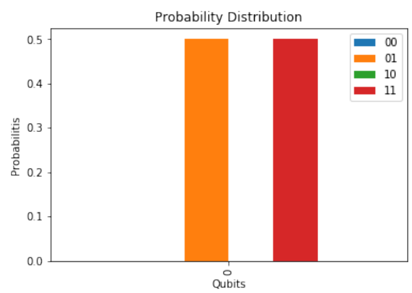

Figure 17: measurement of final state

```
>>> (0.4999999999999998, 2.999999999999999)
```

3.1.6 Alice plays H and Bob plays I

$U_A = H, U_B = I$. so,

$$
\begin{aligned}
|\Psi_s\rangle &= (H \otimes I)\,|\Lambda\rangle & (65)\\
&= \frac{1}{\sqrt{2}}[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle + i\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\,|1\rangle] & (66)\\
&= \frac{1}{\sqrt{2}}[|00\rangle + |10\rangle + i\,|01\rangle - i\,|11\rangle] & (67)
\end{aligned}
$$

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (68)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + |10\rangle + i\,|01\rangle - i\,|11\rangle] && (69)\\
&= \frac{1}{2\sqrt{2}}[|00\rangle - i\,|11\rangle + |10\rangle - i\,|01\rangle + i\,|01\rangle + |10\rangle - i\,|11\rangle - |00\rangle] && (70)\\
&= \frac{1}{2\sqrt{2}}2[|10\rangle - i\,|11\rangle] && (71)\\
&= \frac{1}{\sqrt{2}}[|10\rangle - i\,|11\rangle] && (72)
\end{aligned}
$$

A measurement of the final state yields an equal probability for a payout of $\delta(5)$ or a payout of $\beta(1)$ to Alice and an equal probability for a payout of $\gamma(0)$ or a payout of $\beta(1)$ to Bob. Here, $\overline{\pi_A} = 3$ and $\overline{\pi_B} = 0.5$.

```
QPD(H, I2, 3, 1, 0, 5)
```

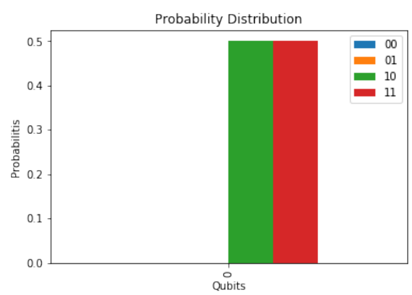

Figure 18: measurement of final state

```
>>> (2.999999999999999, 0.4999999999999998)
```

3.1.7 Alice plays σ_x and Bob plays H

$U_A = \sigma_x, U_B = H$. so,

$$
\begin{aligned}
|\Psi_s\rangle &= (\sigma_x \otimes H)\,|\Lambda\rangle && (73)\\
&= \frac{1}{\sqrt{2}}[|1\rangle \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + i\,|0\rangle \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)] && (74)\\
&= \frac{1}{\sqrt{2}}[|10\rangle + |11\rangle + i\,|00\rangle - i\,|01\rangle] && (75)
\end{aligned}
$$

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (76)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|10\rangle + |11\rangle + i\,|00\rangle - i\,|01\rangle] && (77)\\
&= \frac{1}{2\sqrt{2}}[|10\rangle - i\,|01\rangle + |11\rangle - i\,|00\rangle + i\,|00\rangle + |11\rangle - i\,|01\rangle - |10\rangle] && (78)\\
&= \frac{1}{2\sqrt{2}}2[|11\rangle - i\,|01\rangle] && (79)\\
&= \frac{1}{\sqrt{2}}[|11\rangle - i\,|01\rangle] && (80)
\end{aligned}
$$

A measurement of the final state yields an equal probability for a payout of $\gamma(0)$ or a payout of $\beta(1)$ to Alice and an equal probability for a payout of $\delta(5)$ or a payout of $\beta(1)$ to Bob (look at the payoff matrix to remove confusion). Here, $\overline{\pi_A} = 0.5$ and $\overline{\pi_B} = 3$.

```
sigma_x=PauliX(I2)
H=Hadamard(I2)
QPD(sigma_x, H, 3, 1, 0, 5)
```

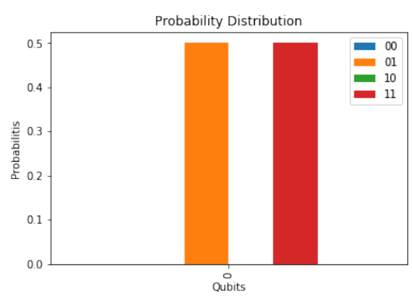

Figure 19: measurement of final state

```
>>> (0.4999999999999998, 2.999999999999999)
```

3.1.8 Alice plays H and Bob plays σ_x

$U_A = H, U_B = \sigma_x$. so,

$$
\begin{aligned}
|\Psi_s\rangle &= (H \otimes \sigma_x)\,|\Lambda\rangle && (81)\\
&= \frac{1}{\sqrt{2}}[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|1\rangle + i\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\,|0\rangle] && (82)\\
&= \frac{1}{\sqrt{2}}[|01\rangle + |11\rangle + i\,|00\rangle - i\,|10\rangle] && (83)
\end{aligned}
$$

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle & (84)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|01\rangle + |11\rangle + i\,|00\rangle - i\,|10\rangle] & (85)\\
&= \frac{1}{2\sqrt{2}}[|01\rangle - i\,|10\rangle + |11\rangle - i\,|00\rangle + i\,|00\rangle + |11\rangle - i\,|10\rangle - |01\rangle] & (86)\\
&= \frac{1}{2\sqrt{2}}2[|11\rangle - i\,|10\rangle] & (87)\\
&= \frac{1}{\sqrt{2}}[|11\rangle - i\,|10\rangle] & (88)
\end{aligned}
$$

A measurement of the final state yields an equal probability for a payout of $\delta(5)$ or a payout of $\beta(1)$ to Alice and an equal probability for a payout of $\gamma(0)$ or a payout of $\beta(1)$ to Bob. Here, $\overline{\pi_A} = 3$ and $\overline{\pi_B} = 0.5$.

```
QPD(H, sigma_x, 3, 1, 0, 5)
```

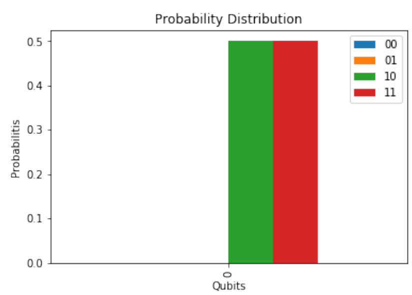

Figure 20: measurement of final state

```
>>> (2.999999999999999, 0.4999999999999998)
```

3.1.9 Both play H

$U_A = H, U_B = H$. so,

$$
\begin{aligned}
|\Psi_s\rangle &= (H \otimes H)\,|\Lambda\rangle & (89)\\
&= (H \otimes H)\frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] & (90)\\
&= \frac{1}{\sqrt{2}}[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + i\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)] & (91)\\
&= \frac{1}{2\sqrt{2}}[|00\rangle + |01\rangle + |10\rangle + |11\rangle + i\,|00\rangle - i\,|01\rangle - i\,|10\rangle + i\,|11\rangle] & (92)
\end{aligned}
$$

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (93)\\
&= \frac{1}{4}(I\otimes I - i\sigma_x\otimes\sigma_x)[|00\rangle + |01\rangle + |10\rangle + |11\rangle + i\,|00\rangle - i\,|01\rangle - i\,|10\rangle + i\,|11\rangle] && (94)\\
&= \frac{1}{4}2[|00\rangle + |11\rangle - i\,|01\rangle - i\,|10\rangle] && (95)\\
&= \frac{1}{2}[|00\rangle + |11\rangle - i\,|01\rangle - i\,|10\rangle] && (96)
\end{aligned}
$$

A measurement of the final state yields an equal probability of 0.25 for a payout of $\alpha(3)$ or a payout of $\gamma(0)$ or a payout of $\delta(5)$ or a payout of $\beta(1)$ to Alice and an equal probability of 0.25 for a payout of $\alpha(3)$ or a payout of $\gamma(0)$ or a payout of $\delta(5)$ or a payout of $\beta(1)$ to Bob. Here, $\overline{\pi_A} = 2.25$ and $\overline{\pi_B} = 2.25$.

```
QPD(H, H, 3, 1, 0, 5)
```

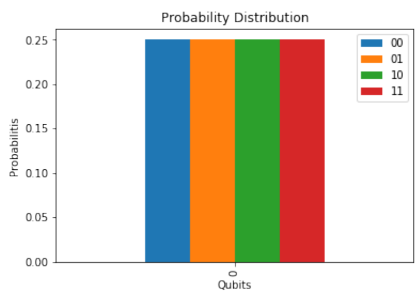

Figure 21: measurement of final state

```
>>> (2.2499999999999987, 2.2499999999999987)
```

3.2 New payoff matrix with allowed quantum moves

We use the software to write a new script that generates the payoff matrices for Alice and Bob separately. We consider $\alpha = 3$, $\beta = 1$, $\gamma = 0$ and $\delta = 5$.

```
def QPD2(U_A, U_B, alpha, beta, gamma, delta):
    sigma_x=PauliX(I2)
    U=(np.kron(I2, I2)+1j*np.kron(sigma_x, sigma_x))/np.sqrt(2)
    U_dag=np.conj(U.T)

    initial=U.dot(Q00)

    PsiS=np.kron(U_A, U_B).dot(initial)
    PsiF=U_dag.dot(PsiS)
```

```
    cpsif=np.conj(PsiF.T)

    def pi(alpha, beta, gamma, delta): return (alpha*np.abs(cpsif.dot(Q00))**2+
    gamma*np.abs(cpsif.dot(Q01))**2+delta*np.abs(cpsif.dot(Q10))**2+
    beta*np.abs(cpsif.dot(Q11))**2, alpha*np.abs(cpsif.dot(Q00))**2+
    delta*np.abs(cpsif.dot(Q01))**2+gamma*np.abs(cpsif.dot(Q10))**2+
    beta*np.abs(cpsif.dot(Q11))**2)

    return pi(alpha, beta, gamma, delta)

moves={1:I2, 2:PauliX(I2), 3:Hadamard(I2), 4:PauliZ(I2)}

def payoff_matrix_1(alpha, beta, gamma, delta):
    Alice=np.zeros([3, 3])
    Bob=np.zeros([3, 3])
    for i in range(1, 4):
        for j in range(1, 4):
            X=QPD2(moves[i], moves[j], alpha, beta, gamma, delta)
            Alice[i-1, j-1], Bob[i-1, j-1]=X[0], X[1]

    return(Alice, Bob)

payoff_matrix_1(3, 1, 0, 5)
```

The output:

```
(array([[3. , 0. , 0.5 ],
        [5. , 1. , 0.5 ],
        [3. , 3. , 2.25]]), array([[3. , 5. , 3. ],
        [0. , 1. , 3. ],
        [0.5 , 0.5 , 2.25]]))
```

We see the payoff matrix generated is:

Alice/ Bob	I	σ_x	H
I	(3, 3)	(0, 5)	(0.5, 3)
σ_x	(5, 0)	(1, 1)	(0.5, 3)
H	(3, 0.5)	(3, 0.5)	(2.25, 2.25)

Using the IDSDS algorithm we see that the *iterated strictly dominant-strategy equilibrium* has changed from $\{\sigma_x, \sigma_x\}$ to $\{H, H\}$ and also $\{\sigma_x, \sigma_x\}$ is no longer the Nash equilibrium but $\{H, H\}$ is.

```
L=payoff_matrix_1(3, 1, 0, 5)
IDSDS(L[0], L[1])
>>> (array([[2.25]]), array([[2.25]]))
```

```
L1=[]
L2=[]

for i in range(3):
    a=L[0][:, i]
    M=max(a)
    for j in range(3):
        if a[j]==M: L1+=[[j, i], ]

for i in range(3):
    a=L[1][i, :]
    M=max(a)
    for j in range(3):
        if a[j]==M: L2+=[[i, j], ]

N=[i for i in L1 if i in L2]

I1={0:'I', 1:'sigma_x', 2:'H'}
I2={0:'I', 1:'sigma_x', 2:'H'}

Nash=[(I1[i], I2[j]) for [i, j] in N]
## The Nash equilibria
Nash
>>> [('H', 'H')]
```

We need to notice that although $(2.25, 2.25)$ corresponding to $\{H, H\}$ is now the strong Nash equilibrium, it is not Pareto optimal. The introduction of quantum moves clearly alters the game's outcome. To induce Pareto optimality in the Nash equilibrium, let's introduce another quantum move σ_z. Now, moves, $S = \{I, \sigma_x, H, \sigma_z\}$.

3.2.1 Alice plays I and Bob plays σ_z

We need to keep in mind that $\sigma_z |0\rangle = |0\rangle$ and $\sigma_z |1\rangle = -|1\rangle$. Here, $U_A = I$ and $U_B = \sigma_z$.

$$|\Psi_s\rangle = (I \otimes \sigma_z) |\Lambda\rangle \tag{97}$$
$$= (I \otimes \sigma_z) U |00\rangle \tag{98}$$
$$= (I \otimes \sigma_z) \frac{1}{\sqrt{2}} [|00\rangle + i |11\rangle] \tag{99}$$
$$= \frac{1}{\sqrt{2}} [|00\rangle - i |11\rangle] \tag{100}$$

So, the final state,

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle & (101)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle - i\,|11\rangle] & (102)\\
&= \frac{1}{2}[|00\rangle - i\,|11\rangle - i\,|11\rangle - |00\rangle] & (103)\\
&= -i\,|11\rangle & (104)
\end{aligned}
$$

We get $|11\rangle$ with probability 1 on measurement of the final state. $\overline{\pi_A} = 1$ and $\overline{\pi_B} = 1$.

```
sigma_z=PauliZ(I2)
QPD(I2, sigma_z, 3, 1, 0, 5)
```

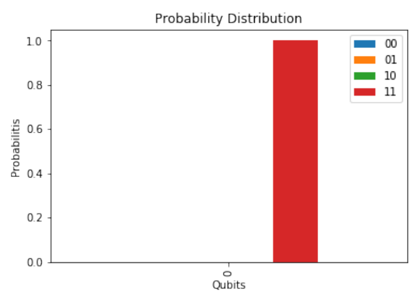

Figure 22: measurement of final state

```
(0.9999999999999996, 0.9999999999999996)
```

3.2.2 Alice plays σ_z and Bob plays I

$U_A = \sigma_z$ and $U_B = I$.

$$
\begin{aligned}
|\Psi_s\rangle &= (\sigma_z \otimes I)\,|\Lambda\rangle & (105)\\
&= (\sigma_z \otimes I)U\,|00\rangle & (106)\\
&= (\sigma_z \otimes I)\frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] & (107)\\
&= \frac{1}{\sqrt{2}}[|00\rangle - i\,|11\rangle] & (108)
\end{aligned}
$$

So, the final state,

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (109)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle - i\,|11\rangle] && (110)\\
&= \frac{1}{2}[|00\rangle - i\,|11\rangle - i\,|11\rangle - |00\rangle] && (111)\\
&= -i\,|11\rangle && (112)
\end{aligned}$$

We get $|11\rangle$ with probability 1 on measurement of the final state. $\overline{\pi_A} = 1$ and $\overline{\pi_B} = 1$.

```
sigma_z=PauliZ(I2)
QPD(sigma_z, I2, 3, 1, 0, 5)
```

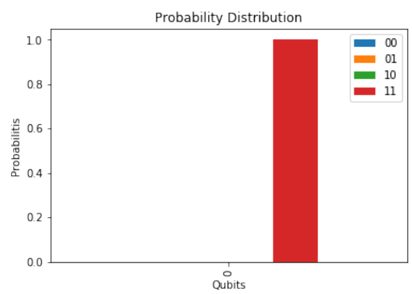

Figure 23: measurement of final state

3.2.3 Alice plays σ_x and Bob plays σ_z

$U_A = \sigma_x$ and $U_B = \sigma_z$.

$$\begin{aligned}
|\Psi_s\rangle &= (\sigma_z \otimes \sigma_z)\,|\Lambda\rangle && (113)\\
&= (\sigma_x \otimes \sigma_z)U\,|00\rangle && (114)\\
&= (\sigma_x \otimes \sigma_z)\frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] && (115)\\
&= \frac{1}{\sqrt{2}}[|10\rangle - i\,|01\rangle] && (116)
\end{aligned}$$

So, the final state,

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (117)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|10\rangle - i\,|01\rangle] && (118)\\
&= \frac{1}{2}[|10\rangle - i\,|01\rangle - i\,|01\rangle - |10\rangle] && (119)\\
&= -i\,|01\rangle && (120)
\end{aligned}$$

We get $|01\rangle$ with probability 1 on measurement of the final state. $\overline{\pi_A} = 0$ and $\overline{\pi_B} = 5$.

```
QPD(sigma_x, sigma_z, 3, 1, 0, 5)
```

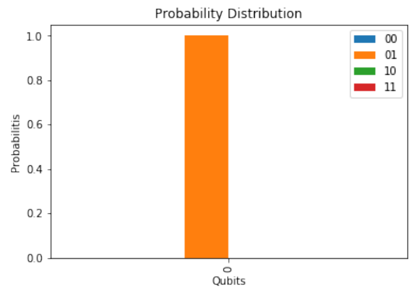

Figure 24: measurement of final state

```
(0.0, 4.999999999999998)
```

3.2.4 Alice plays σ_z and Bob plays σ_x

$U_A = \sigma_z$ and $U_B = \sigma_x$.

$$\begin{aligned}
|\Psi_s\rangle &= (\sigma_z \otimes \sigma_x) |\Lambda\rangle && (121)\\
&= (\sigma_z \otimes \sigma_x) U |00\rangle && (122)\\
&= (\sigma_z \otimes \sigma_x) \frac{1}{\sqrt{2}}[|00\rangle + i |11\rangle] && (123)\\
&= \frac{1}{\sqrt{2}}[|01\rangle - i |10\rangle] && (124)
\end{aligned}$$

So, the final state,

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle && (125)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|01\rangle - i |10\rangle] && (126)\\
&= \frac{1}{2}[|01\rangle - i |10\rangle - i |10\rangle - |01\rangle] && (127)\\
&= -i |10\rangle && (128)
\end{aligned}$$

We get $|10\rangle$ with probability 1 on measurement of the final state. $\overline{\pi_A} = 5$ and $\overline{\pi_B} = 0$.

```
QPD(sigma_z, sigma_x, 3, 1, 0, 5)
```

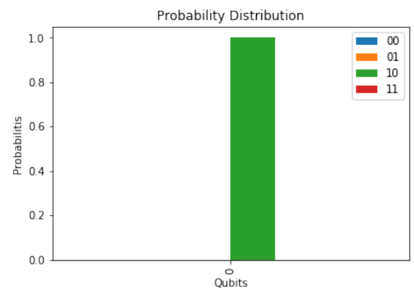

Figure 25: measurement of final state

```
(4.999999999999998, 0.0)
```

3.2.5 Alice plays σ_z and Bob plays H

$U_A = \sigma_z$ and $U_B = H$.

$$\begin{aligned}
|\Psi_s\rangle &= (\sigma_z \otimes H)\,|\Lambda\rangle && (129)\\
&= (\sigma_z \otimes H) U\,|00\rangle && (130)\\
&= (\sigma_z \otimes H)\frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] && (131)\\
&= \frac{1}{\sqrt{2}}[|0\rangle\,\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) - i\,|1\rangle\,\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)] && (132)\\
&= \frac{1}{\sqrt{2}}[|00\rangle + |01\rangle - i\,|10\rangle + i\,|11\rangle] && (133)
\end{aligned}$$

So, the final state,

$$\begin{aligned}
|\Psi_f\rangle &= U^\dagger\,|\Psi_s\rangle && (134)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + |01\rangle - i\,|10\rangle + i\,|11\rangle] && (135)\\
&= \frac{1}{2\sqrt{2}}[|00\rangle - i\,|11\rangle + |01\rangle - i\,|10\rangle - i\,|10\rangle - |01\rangle + i\,|11\rangle + |00\rangle] && (136)\\
&= \frac{1}{2}[|00\rangle - i\,|10\rangle] && (137)
\end{aligned}$$

A measurement of the final state yields an equal probability of 0.5 for a payout of $\alpha(3)$ or a payout of $\delta(5)$ to Alice and an equal probability of 0.5 for a payout of $\alpha(3)$ or a payout of $\gamma(0)$ to Bob. Here, $\overline{\pi_A} = 4$ and $\overline{\pi_B} = 1.5$.

```
QPD(sigma_z, H, 3, 1, 0, 5)
```

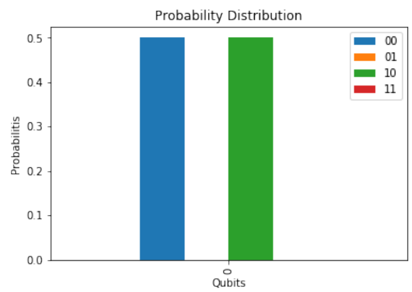

Figure 26: measurement of final state

```
(3.9999999999999982, 1.4999999999999993)
```

3.2.6 Alice plays H and Bob plays σ_z

$U_A = H$ and $U_B = \sigma_z$.

$$
\begin{aligned}
|\Psi_s\rangle &= (H \otimes \sigma_z)\,|\Lambda\rangle && (138)\\
&= (H \otimes \sigma_z)U\,|00\rangle && (139)\\
&= (H \otimes \sigma_z)\frac{1}{\sqrt{2}}[|00\rangle + i\,|11\rangle] && (140)\\
&= \frac{1}{\sqrt{2}}[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|0\rangle - i\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\,|1\rangle] && (141)\\
&= \frac{1}{2}[|00\rangle + |10\rangle - i\,|01\rangle + i\,|11\rangle] && (142)
\end{aligned}
$$

So, the final state,

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger\,|\Psi_s\rangle && (143)\\
&= \frac{1}{2\sqrt{2}}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + |10\rangle - i\,|01\rangle + i\,|11\rangle] && (144)\\
&= \frac{1}{2\sqrt{2}}[|00\rangle - i\,|11\rangle + |10\rangle - i\,|01\rangle - i\,|01\rangle - |10\rangle + i\,|11\rangle + |00\rangle] && (145)\\
&= \frac{1}{\sqrt{2}}[|00\rangle - i\,|01\rangle] && (146)
\end{aligned}
$$

A measurement of the final state yields an equal probability of 0.5 for a payout of $\alpha(3)$ or a payout of $\gamma(0)$ to Alice and an equal probability of 0.5 for a payout of $\alpha(3)$ or a payout of $\delta(5)$ to Bob. Here, $\overline{\pi_A} = 1.5$ and $\overline{\pi_B} = 4$.

```
QPD(H, sigma_z, 3, 1, 0, 5)
```

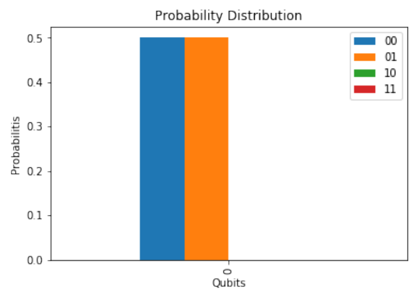

Figure 27: measurement of final state

```
(1.4999999999999993, 3.9999999999999982)
```

3.2.7 Both play σ_z

$U_A = \sigma_z$ and $U_B = \sigma_z$.

$$
\begin{aligned}
|\Psi_s\rangle &= (\sigma_z \otimes \sigma_z) |\Lambda\rangle &(147)\\
&= (\sigma_z \otimes \sigma_z) U |00\rangle &(148)\\
&= (\sigma_z \otimes \sigma_z) \frac{1}{\sqrt{2}} [|00\rangle + i |11\rangle] &(149)\\
&= \frac{1}{\sqrt{2}} [|00\rangle + i |11\rangle] &(150)
\end{aligned}
$$

So, the final state,

$$
\begin{aligned}
|\Psi_f\rangle &= U^\dagger |\Psi_s\rangle &(151)\\
&= \frac{1}{2}(I \otimes I - i\sigma_x \otimes \sigma_x)[|00\rangle + i |11\rangle] &(152)\\
&= \frac{1}{2}[|00\rangle - i |11\rangle + i |11\rangle + |00\rangle] &(153)\\
&= |00\rangle &(154)
\end{aligned}
$$

We get $|00\rangle$ with probability 1 on measurement of the final state. $\overline{\pi_A} = 3$ and $\overline{\pi_B} = 3$.

```
QPD(sigma_z, sigma_z, 3, 1, 0, 5)
```

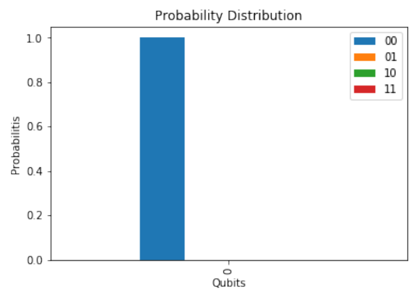

Figure 28: measurement of final state

```
(2.9999999999999987, 2.9999999999999987)
```

3.3 The final payoff matrix with introduction of another quantum move

```
def payoff_matrix_2(alpha, beta, gamma, delta):
    Alice=np.zeros([4, 4])
    Bob=np.zeros([4, 4])
    for i in range(1, 5):
        for j in range(1, 5):
            X=QPD2(moves[i], moves[j], alpha, beta, gamma, delta)
            Alice[i-1, j-1], Bob[i-1, j-1]=X[0], X[1]

    return(Alice, Bob)

payoff_matrix_2(3, 1, 0, 5)
```

The output:

```
(array([[3. , 0. , 0.5 , 1. ],
        [5. , 1. , 0.5 , 0. ],
        [3. , 3. , 2.25, 1.5 ],
        [1. , 5. , 4. , 3. ]]), array([[3. , 5. , 3. , 1. ],
        [0. , 1. , 3. , 5. ],
        [0.5 , 0.5 , 2.25, 4. ],
        [1. , 0. , 1.5 , 3. ]]))
```

We see the payoff matrix generated is:

Alice/ Bob	I	σ_x	H	σ_z
I	(3, 3)	(0, 5)	(0.5, 3)	(1, 1)
σ_x	(5, 0)	(1, 1)	(0.5, 3)	(0, 5)
H	(3, 0.5)	(3, 0.5)	(2.25, 2.25)	(1.5, 4)
σ_z	(1, 1)	(5, 0)	(4, 1.5)	(3, 3)

Using the IDSDS algorithm we see that the *iterated strictly dominant-strategy equilibrium* has changed from $\{H, H\}$ to $\{\sigma_z, \sigma_z\}$ and also $\{H, H\}$ is no longer the Nash equilibrium but $\{\sigma_z, \sigma_z\}$ is.

```
L=payoff_matrix_2(3, 1, 0, 5)
IDSDS(L[0], L[1])
>>> (array([[3]]), array([[3]]))
```

```
L1=[]
L2=[]

for i in range(4):
    a=L[0][:, i]
    M=max(a)
    for j in range(4):
        if a[j]==M: L1+=[[j, i], ]

for i in range(4):
    a=L[1][i, :]
    M=max(a)
    for j in range(4):
        if a[j]==M: L2+=[[i, j], ]

N=[i for i in L1 if i in L2]

I1={0:'I', 1:'sigma_x', 2:'H', 3: 'sigma_z'}
I2={0:'I', 1:'sigma_x', 2:'H', 3: 'sigma_z'}

Nash=[(I1[i], I2[j]) for [i, j] in N]
## The Nash equilibria
Nash
>>> [('sigma_z', 'sigma_z')]
```

We need to notice that $(3, 3)$ corresponding to $\{\sigma_z, \sigma_z\}$ is now not only the Nash equilibrium, but also is Pareto optimal.

4 Conclusion

This project conveys the message that a simple quantum program can be easily designed in any programming language (Python, Matlab, R) by anyone to start simulating quantum phenomenon in a classical computer. Students from diverse backgrounds like physics, economics and computer science can carry out further implementations in designing some interesting models.

5 Acknowledgement

I would like to express my special thanks of gratitude to my Professors, Dr. Debasish Lohar and Dr. Subhankar Ray who helped me with their useful suggestions to complete this project in time.

References

[1] J. Orlin Grabbe, *"An Introduction to Quantum Game Theory"*, https://arxiv.org/abs/quant-ph/0506219 v1 Mon, 27 Jun 2005

[2] Giacomo Bonanno, *Game Theory (Open Access textbook with 165 solved exercises)*, https://arxiv.org/abs/1512.06808 v1 Mon, 21 Dec 2015

[3] M. A. Nielsen and I. L. Chuang, *"Quantum computation and quantum information"* (Cambridge University Press,2000)

[4] IBM Q Experience, https://quantumexperience.ng.bluemix.net/qx

[5] Indranil Ghosh, *"Quantum-SimuPy"*, https://github.com/indrag49/Quantum-SimuPy

[6] Indranil Ghosh, *"Scipy-2018-India"*, https://github.com/indrag49/Scipy-2018-India