AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

KATEDRA INFORMATYKI

PRACA DYPLOMOWA MAGISTERSKA

Quantum games on IBM-Q

Gry kwantowe na IBM-Q

Autor: *Filip Galas*
Kierunek studiów: informatyka
Typ studiów: *Stacjonarne*
Opiekun pracy: *dr inż. Katarzyna Rycerz*

Kraków, 2019

Oświadczenie studenta

Uprzedzony(-a) o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2018 r. poz. 1191 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony(-a) o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta.”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Jednocześnie Uczelnia informuje, że zgodnie z art. 15a ww. ustawy o prawie autorskim i prawach pokrewnych Uczelni przysługuje pierwszeństwo w opublikowaniu pracy dyplomowej studenta. Jeżeli Uczelnia nie opublikowała pracy dyplomowej w terminie 6 miesięcy od dnia jej obrony, autor może ją opublikować, chyba że praca jest częścią utworu zbiorowego. Ponadto Uczelnia jako podmiot, o którym mowa w art. 7 ust. 1 pkt 1 ustawy z dnia 20 lipca 2018 r. – Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.), może korzystać bez wynagrodzenia i bez konieczności uzyskania zgody autora z utworu stworzonego przez studenta w wyniku wykonywania obowiązków związanych z odbywaniem studiów, udostępniać utwór ministrowi właściwemu do spraw szkolnictwa wyższego i nauki oraz korzystać z utworów znajdujących się w prowadzonych przez niego bazach danych, w celu sprawdzania z wykorzystaniem systemu antyplagiatowego. Minister właściwy do spraw szkolnictwa wyższego i nauki może korzystać z prac dyplomowych znajdujących się w prowadzonych przez niego bazach danych w zakresie niezbędnym do zapewnienia prawidłowego utrzymania i rozwoju tych baz oraz współpracujących z nimi systemów informatycznych.

……

(czytelny podpis studenta)

STRESZCZENIE

Technologia obliczeń kwantowych przeżywa obecnie gwałtowny rozwój. W 2016 roku udostępniono w chmurze pierwszy komputer kwantowy w ramach projektu IBM Q [1], co ma w założeniach doprowadzić do zainteresowania się dziedziną obliczeń kwantowych szersze grono naukowców i studentów. Od tamtego czasu udostępniane są nowe narzędzia i urządzenia kwantowe o coraz lepszej wydajności [2, 3]. Wydaje się zasadne pytanie, czy dziedzina gier kwantowych, rozważana teoretycznie już od czasów pracy D. A. Meyera [4], a obecnie znajdująca nowe zastosowania [5, 6], ma szanse na eksperymentalną realizację swoich osiągnięć na obecnych komputerach kwantowych z serii IBM Q. Ich uniwersalność a także wysoka dostępność stanowi doskonałe warunki do badań nad grami kwantowymi. Niniejsza praca przedstawia pierwszą realizację kwantowego dylematu więźnia w schemacie Eiserta, Wilkensa i Lewensteina [7] na komputerze kwantowym IBM Q. Opracowano postać schematu EWL, która może być uruchamiana na komputerach IBM Q, zbadano wpływ błędów urządzeń kwantowych na wyniki gry, a także zastosowano metody kwantowej korekcji błędów w celu poprawy tych wyników.

SUMMARY

Quantum computing technology is currently undergoing rapid development. In 2016, the first quantum computer of the IBM Q project was made available in the cloud [1], which was expected to lead to a wider interest in the field of quantum computing among scientists and students. Since then, new quantum tools and devices with better and better performance have been made available [2, 3]. It seems reasonable to ask whether the field of quantum games, considered theoretically since the work of D. A. Meyer [4], and now finding new applications [5, 6], has a chance for experimental implementation of its achievements on the current quantum computers from the IBM Q series. Their universality and high availability provide excellent conditions for research on quantum games. This paper presents the first realization of the quantum Prisoner's Dilemma in the Eisert, Wilkens and Lewenstein scheme [7] on the IBM Q quantum computer. A form of EWL scheme which meets the requirements of IBM Q computers has been developed, the impact of quantum device errors on game results has been studied, and quantum error correction methods have been applied to improve these results.

I would like to express my great appreciation to my supervisor dr inż. Katarzyna Rycerz for her constant support during writing this thesis. I also would like to thank Michał Śmietana and my Mother for verifying linguistic corectness.

Contents

1. Introduction

1.1. A short introduction to quantum games

The term *quantum game* comes from the work of Eisert, Wilkens, and Lewenstein [7], in which it appears to have been first used [8]. It consists of two parts indicating two distinct fields of knowledge in which the concept of quantum games is rooted.

Firstly, a quantum game is a *game* and thus it can be understood and theoretized using concepts known from the game theory — it consists of a set of strategies for each player and a payoff function [9]. We can study them in order to find strategies which rational players would choose, ex. a Nash equilibrium, which is such a joint strategy that "no player can achieve a higher payoff by *unilaterally* switching to another strategy." ([9])

However, there is also the second part of the term, the *quantum* part, which relates to the quantum information theory. This has multiple implications. First of all, it means that the state of a quantum game is represented as a *qubit* or a register of qubits. A qubit is the unit of quantum information which can be in any linear combination of two states $|0\rangle$, $|1\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle\,, \qquad \text{where } \alpha, \beta \in \mathbb{C} \text{ and } |\alpha|^2 + |\beta|^2 = 1\,, \tag{1.1}$$

as opposed to classical bits, which can only be in one of two states: 0 or 1 [10]. Furthermore, players execute their strategies by applying unitary operations (*quantum gates*) to influence the quantum state of the game. Lastly, the outcome of the game is determined by the act of measuring the quantum state and applying the payoff function to the measured result in order to obtain the players' payoffs.

It was D. A. Meyer [4] who came up with the idea of using *quantum strategies*, as he called them, in a game. He introduced a simple game called *Penny Flip* (which he very attractively presented as a duel between Captain Picard and Q — two characters from a famous television series *Star Trek: The Next Generation*) and showed that one player could always win (it was Q in his narrative) if he was allowed to use quantum strategies against the other player who knew nothing about them (poor Picard), thus was restricted to the familiar classical strategies. Since then a number of quantum games have been proposed [7, 11, 12, 13] with intriguing properties. It is also worth mentioning that this problem is not entirely theoretical as applications are being found [5, 6]. Since not long after the conception of quantum games there have been attempts to realize them on experimental quantum computers [12, 13, 14, 15, 16, 17].

1.2. Motivation

As universal quantum computing technology has been made open to the public by IBM through their *IBM Q Experience* platform [18], the question of their capabilities in the field of quantum game realization is still left unanswered.

Previous experimental realizations of quantum games [12, 13, 14, 15, 16] have their limitations. The quantum hardware used to perform the experiments is not publicly accessible, therefore their results are not easily reproducible. Moreover, in the case of the linear optics implementations [12, 13, 15, 16], the construction of a large number of strategies is infeasible.

However, the quantum computers from the IBM Q project could overcome those limitations, as they are publicly accessible through the Internet and they implement the universal one-qubit quantum gates as well as their controlled counterparts (in fact this allows the construction of any multi-qubit quantum gate [19]).

For these reasons, this thesis will attempt to accomplish a complete (ie. including all quantum strategies) realization of a quantum game which would be playable on an IBM Q quantum computer over the Internet.

1.3. Related work

This section discusses a selection of papers concerning quantum games themselves as well as experimental realizations.

1.3.1. Quantum games

The theoretical background of this work is based on two aforementioned papers of major significance in the quantum game field.

The first one is the Meyer's work [4]. Meyer introduces a two-person zero-sum game (*Penny Flip*), in which the state of the game is represented as a vector from a two-dimensional complex Hilbert's space. He shows that the player who utilizes *quantum* strategies has an advantage over the player restricted to *mixed classical* strategies (see Theorem 2.3.2 in Chapter 2).

Eisert, Wilkens, and Lewenstein further this idea in their paper [7], where they present a quantum generalization of a two-person nonzero-sum game known as the *Prisoner's Dilemma*. They propose a physical model of the game based on its quantum formulation, which will be called the *EWL protocol* or the *EWL scheme* throughout this work after its originators (this is an important result for this work, as it will be focused on the EWL scheme implementation). Then they discuss the influence of the amount of entanglement induced by the entangling gate on the outcome of the game. Authors also show that there is a quantum strategy for this game that always gives a reward against any classical strategy.

1.3.2. Experimental realizations

When it comes to experimental realizations of quantum games there is a lot of published papers on this subject, therefore only a selection of them will be presented with the focus on the EWL Prisoner's Dilemma scheme's implementations.

Du et al. managed to implement the EWL protocol on their nuclear magnetic resonance quantum computer [14]. They discuss the game structure as the entanglement parameter γ (which is responsible for the amount of entanglement the players' qubits are subjected to) varies from 0 (separable game) to $\pi/2$ (maximum entanglement) and present payoff value plots for three different values of γ as a function of players' strategies. The theoretic and experimental plots of the expected payoff as a function of the parameter γ (assuming the players resort to Nash equilibrium) are then compared.

Prevedel et al. in their work [15] "report the first demonstration of a quantum game on an all-optical one-way quantum computer." The implemented game is a quantum version of the Prisoner's Dilemma which, however, differs from the EWL protocol by using a different entangling gate — Prevedel et al. use a combination of Hadamard and CPhase operations. They construct four strategies for each player (two classical — *cooperate* and *defect* and two quantum strategies). Experimental payoffs for the implemented strategies are then compared to the theoretical payoff function plot.

Pinheiro et al. implement the quantum Prisoner's Dilemma in the EWL scheme using linear optics [16]. Only the maximally entangled version of the game is considered. Five different strategies (*cooperate*, *defect*, and three quantum strategies) for each player are constructed. In conclusion the theoretical payoff function plot is presented along with the points corresponding to the values obtained in the experiment.

Zhang et al. propose a model for a quantum gambling machine [12] and prove its feasibility by demonstrating its optical circuit.

Balthazar et al. realized a quantum duel game using linear optics [13] and show how the game is generalized from its classical counterpart.

There are already a few quantum game implementations on IBM Q [20] (although it should be noted that not all games cited in the reference are quantum games in our sense). One of the most significant is the *Quantum Coin Game* [17], as it is an implementation of the Meyer's quantum coin flip game [4]. However, there are not yet any published realizations of the quantum Prisoner's Dilemma on IBM Q and therefore this will be the basis of this work.

1.4. Objectives

The main objective of this thesis is to assess the possibility of realizing the EWL Scheme on IBM Q. This is achieved by the following goals:

Quantum game realization on IBM Q

The first objective is to realize a quantum game, namely the quantum Prisoner's Dilemma as described in [7] (EWL scheme), on an IBM Q quantum computer.

In order to achieve this goal the quantum circuit of the EWL scheme needs to be formulated by using quantum gates available on IBM Q computers. This means that the two-qubit entangling and disentangling gates of the scheme require decomposition, as IBM Q does not allow universal multi-qubit gates [21].

After the decomposition has been made, the quantum circuit of the EWL scheme should be implemented in Python using the IBM Q's library for quantum computation: *qiskit* [3].

Study on the influence of the quantum errors

All quantum devices suffer from their physical limitations. For example, a phenomenon called *quantum decoherence* [10] is known to occur on these devices, which lead to the corruption of quantum states. Moreover, the act of measurement itself is not ideal and introduces a non-negligible error [10].

Therefore, the second objective of this work is to study the influence of the quantum computer errors on the quantum game results.

Quantum error correction

An important aspect of quantum computation is *quantum error correction* [10]. In fact, it has been shown that there are methods which allow the mitigation of the quantum errors mentioned above [10].

The third objective is then to use and evaluate some of these methods.

1.5. Structure of the work

Chapter 2 provides a theoretical background to quantum games. It starts by introducing basic concepts of quantum computation as well as game theory. Then the D. A. Meyer's Penny Flip quantum game is discussed and on this example some characteristics of quantum games are given. The chapter concludes with a thorough discussion on the Prisoner's Dilemma — its classical version as well as the quantum generalization, namely the EWL scheme, which is the main focus of this thesis.

Chapter 3 discusses issues regarding the realization of quantum algorithms on the IBM Q quantum computers. It starts with an introduction to IBM Q and then proceeds to discuss the topics of quantum errors and quantum error correction. The chapter concludes with a short section concerning circuit optimization on IBM Q.

Chapter 4 presents the solution for the problems stated in this chapter. It starts with the methodology and then proceeds to discuss the EWL scheme realization and ends with a section on the application of quantum error correction methods.

Chapter 5 presents the evaluation of the solution given in Chapter 4. It starts by assessment of the validity of the EWL realization itself and then proceeds to evaluate the quantum error influence on the game results as well as the efficiency of the quantum error correction methods.

Chapter 6 concludes this thesis with a summary of the achieved goals and some propositions regarding future works in this area.

2. Quantum games

This chapter gives a theoretical background to quantum games. It starts by introducing basic concepts of quantum computation and game theory and then proceeds to introduce the D. A. Meyer's Penny Flip game and discuss the characteristics of quantum games on this example. Then the Prisoner's Dilemma game and its quantum generalization, namely the EWL scheme, is discussed.

2.1. Quantum computation

We will begin this chapter with an introduction to quantum computation in general. It will provide theoretical foundations, which will be helpful in understanding quantum games.

2.1.1. Quantum states

A quantum state is described as an element from a projective Hilbert space $\mathbb{C}\mathcal{P}^{n-1}$ of a n-dimensional complex Hilbert space $\mathcal{H}_n$. $\mathbb{C}\mathcal{P}^{n-1}$ is the quotient set of $\mathcal{H}_n$ for the equivalence relation $\sim$ defined as $|\psi\rangle \sim |\phi\rangle \iff \exists c \in \mathbb{C} \quad |\psi\rangle = c|\phi\rangle$. It is conventional to choose normalized vectors $|\psi\rangle \in \mathcal{H}_n$ (ie. such $|\psi\rangle$ that $\langle\psi|\psi\rangle = 1$) as representatives for elements from $\mathbb{C}\mathcal{P}^{n-1}$ (called *rays*).

An important space for quantum computation is $\mathbb{C}\mathcal{P}^1$, which is the state space of a *qubit* (also called the *Bloch sphere*). If $B = \{|0\rangle, |1\rangle\}$ is a basis for $\mathcal{H}_2$, we can write down the state of a qubit as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \qquad \text{for some } \alpha, \beta \in \mathbb{C}\,, \text{ where } |\alpha|^2 + |\beta|^2 = 1\,. \tag{2.1}$$

2.1.2. State evolution

The evolution of a quantum state in time is determined by a unitary operator, ie. such an operator U that

$$U^\dagger U = U U^\dagger = I\,, \tag{2.2}$$

where $U^\dagger$ represents the Hermitian conjugate of U and I is the identity operator.

Notable examples of unitary operators in quantum computing are (given along with their matrix representations in $\mathcal{H}_2$):

– The identity operator $I \equiv \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

– The Pauli operators $X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $Y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$, $Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$.

– The Hadamard gate $H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$.

2.1.3. Measurement

A measurement is defined with respect to an *observable* M, ie. a Hermitian operator on the observed state space. M has a spectral decomposition

$$M = \sum_i a_i P_i \,, \tag{2.3}$$

where P_i is the projector onto the eigenspace of M spanned by eigenvector $|i\rangle$ with eigenvalue a_i. The probability of obtaining outcome a_i is given by

$$p(a_i) = \langle \psi | P_i | \psi \rangle \,. \tag{2.4}$$

After the measurement, if the measured value was a_i, the quantum state collapses to

$$\frac{P_i(\psi)}{\sqrt{p(a_i)}} \,. \tag{2.5}$$

The expectation value of a measurement with respect to an observable M is given by

$$\langle M \rangle = \sum_i a_i \langle \psi | P_i | \psi \rangle = \langle \psi | M | \psi \rangle \,. \tag{2.6}$$

A very important type of measurement is the one with respect to the Pauli operator Z, which is also called the *measurement in the computational basis*. In this type of measurement, the quantum state will collapse to $|0\rangle$ with probability $|\alpha|^2$ and to $|1\rangle$ with probability $|\beta|^2$. Unless stated otherwise, all measurements throughout this thesis are performed in the computational basis.

2.1.4. Mixed quantum states

In the description of quantum states above we talked about *pure*, ie. well-determined, quantum states. However, it is also possible to consider classical probabilistic mixtures of quantum states called *mixed* quantum states. This is the case when, for example, we know that there is a 50/50 chance that a qubit will be in a certain pure quantum state or another. Note that this is very much different from a situation, where a measurement of a qubit in a pure state will yield 0 or 1 with equal probability, because the qubit is in fact in a well-determined physical state.

It turns out that we can generalize the above description of pure quantum states to also include the classical mixtures. There is a way of representing quantum states, which makes it very convenient,

namely the *density operator* or *density matrix* representation. The density operator ρ for a mixed quantum state which is in a pure quantum state $|\psi_i\rangle$ with probability p_i is given by

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \,. \tag{2.7}$$

Then the expectation value of a measurement with respect to an observable M is given by

$$\langle M\rangle = \sum_i p_i \langle\psi_i|M|\psi_i\rangle = \mathrm{Tr}(\rho M) \,. \tag{2.8}$$

2.1.5. Quantum circuits

The state of a quantum computer can be the state of a single qubit, but more commonly it is composed of multiple qubits. The composition of quantum states is the tensor product of component spaces, for example the state space of two qubits is given by $\mathbb{C}\mathcal{P}^1 \otimes \mathbb{C}\mathcal{P}^1$, where $\otimes$ denotes the tensor product.

Generally speaking, a quantum computation Q is a function which transforms a quantum state:

$$Q : \otimes_{i=1}^n \mathbb{C}\mathcal{P}^m \to \otimes_{i=1}^n \mathbb{C}\mathcal{P}^m \quad \text{for some } n, m \in \mathbb{N} \,. \tag{2.9}$$

Quantum algorithms are often visualized as quantum circuit diagrams, which are analogous to their classical counterparts (for examples of quantum circuit diagrams see Figures 2.1, 2.2, and 4.1 below). The flow of time in a quantum circuit diagram is from left to right. Qubit's states are graphically represented as lines (just as wires connect points with the same electric potential in classical circuit diagrams), quantum gates are depicted as rectangles which are connected to those qubits to which there are applied (just as logic gates are connected to wires), and measurements are also included as rectangles with a special symbol.

2.2. Basics of game theory

Game theory is a branch of mathematics which study mathematical models of interactions, involving conflict as well as cooperation, between rational decision-makers who are interested in maximizing their profits. Achievements of game theory are widely used in social sciences such as economics, psychology, and politics to analyze the behavior of social groups, especially in situations of conflict [22]. As we will also use them in our analysis of quantum games, it is necessary to recall some of the basic concepts of game theory.

The key idea of game theory is of course the concept of a *game*. We will be interested in a model of a game called the *normal form*.

Definition 2.2.1 (Normal-form game [9, 23])**.** An n-person **normal-form game** is a tuple (N, S, p), where:

– N is a finite set of n **players**, indexed by i;

- $S = S_1 \times \ldots \times S_n$, where S_i is a set of **strategies** for player i. If S is finite, the game is called finite. An element $s = (s_1, \ldots, s_n) \in S$ is called a **strategy profile** or **joint strategy**. The sequence $(s_j)_{j \neq i}$ will be abbreviated to s_{-i} and s will be sometimes written as (s_i, s_{-i});
- $p = (p_1, \ldots, p_n)$, where $p_i : S \to \mathbb{R}$ is a **payoff function** for player i.

We will also use the following definitions in our analysis of rational decisions:

Definition 2.2.2 (Best response [9])**.** A strategy s_i of player i is a **best response** to a joint strategy s_{-i} of his opponents if

$$\forall s'_i \in S_i \; p_i(s_i, s_{-i}) \geq p_i(s'_i, s_{-i}) \,. \tag{2.10}$$

Definition 2.2.3 (Nash equilibrium [9])**.** A strategy profile $s = (s_1, \ldots, s_n)$ is a **Nash equilibrium** if each s_i is a best response to s_{-i}, that is, if

$$\forall i \in \{1, \ldots, n\} \; \forall s'_i \in S_i \; p_i(s_i, s_{-i}) \geq p_i(s'_i, s_{-i}) \,. \tag{2.11}$$

Definition 2.2.4 (Domination [9])**.** A strategy s_i of player i **dominates** another strategy s'_i of that player if

$$\forall s_{-i} \in S_{-i} \; p_i(s_i, s_{-i}) \geq p_i(s'_i, s_{-i}) \,. \tag{2.12}$$

Definition 2.2.5 (Strict domination [9])**.** A strategy s_i of player i **strictly dominates** another strategy s'_i of that player if

$$\forall s_{-i} \in S_{-i} \; p_i(s_i, s_{-i}) > p_i(s'_i, s_{-i}) \,. \tag{2.13}$$

Definition 2.2.6 ((Strictly) dominant strategy [9])**.** A strategy s_i of player i is **(strictly) dominant** if it (strictly) dominates all other strategies of that player.

Definition 2.2.7 (Pareto efficiency [9])**.** A strategy profile s is **Pareto efficient** if for no strategy profile s'

$$\forall i \in \{1, \ldots, n\} \; p_i(s') \geq p_i(s) \quad \text{and} \quad \exists i \in \{1, \ldots, n\} \; p_i(s') > p_i(s) \,. \tag{2.14}$$

An important extension to ordinary strategies as defined above are probabilistic *mixed strategies*.

Definition 2.2.8 (Mixed strategy [9])**.** A probability distribution m_i on a set of strategies S_i for player i, ie. a function $m_i : S_i \to [0, 1]$ such that $\sum_{s_i \in S_i} m_i(s_i) = 1$, is called a **mixed strategy** of that player. The set of probability distributions on S_i will be denoted by ΔS_i and called the set of mixed strategies for player i. Each strategy $s_i \in S_i$ can be identified with m_i such that $m_i(s_i) = 1$ — these strategies will also be called **pure strategies**.

Definition 2.2.9 (Mixed strategy profile [9])**.** An element $m = (m_1, \ldots, m_n) \in M = \Delta S_1 \times \ldots \times \Delta S_n$ is called a **mixed strategy profile**. We define that for each $s \in S$

$$m(s) := m_1(s_1) \cdot \ldots \cdot m_n(s_n) \,, \tag{2.15}$$

thus m is also a probability distribution on strategy profiles.

Definition 2.2.10 (Expected payoff function [9])**.** A function $E(p_i) : M \to \mathbb{R}$ is called an **expected payoff function** for player i if

$$\forall m \in M \ (E(p_i))(m) = \sum_{s \in S} m(s) \cdot p_i(s) \,. \tag{2.16}$$

Definition 2.2.11 (Probabilistic Nash equilibrium [9])**.** A mixed strategy profile $m = (m_1, \ldots, m_n)$ is a **probabilistic Nash equilibrium** or **Nash equilibrium on mixed strategies** if

$$\forall i \in \{1, \ldots, n\} \ \forall m_i' \in \Delta S_i \ (E(p_i))(m_i, m_{-i}) \geq (E(p_i))(m_i', m_{-i}) \,. \tag{2.17}$$

2.3. The Penny Flip game

Having introduced some basic concepts of quantum computing and game theory we are now prepared to enter the realm of quantum games by discussing the D. A. Meyer's seminal work [4] and his *Penny Flip* game.

The game is described as a duel between Captain Picard of the Starship Enterprise and a strange entity called Q.[1] Apparently, the Enterprise is in danger and Q agreed to help under one circumstance: that Picard beats him in a simple game involving a coin (penny). The game is played as follows:

1. **The penny** is placed **heads up** in a box (in such a way as to obscure its state to the players).
2. **Q** can **flip the coin** in the box without telling his decision to Picard.
3. **Picard** has now the option to **flip the coin** (his decision also remains unknown to Q).
4. **Q** can **flip the coin** again (not telling Picard as before).
5. The box is opened and players can **see** the arrangement of the coin. **Q wins** if the penny is **heads up**. Otherwise, **Picard wins**.

2.3.1. The classical Penny Flip (as Picard sees it)

As Captain Picard is an intelligent man and also has finished a game theory course during his training at Starfleet Academy, he can easily analyze this game using game-theoretical concepts and decide on his optimal strategy. A good starting point seems to be to write down the *payoff matrix* for the Penny Flip game (see Table 2.1 below).

	$N\,N$	$N\,F$	$F\,N$	$F\,F$
N	-1	1	1	-1
F	1	-1	-1	1

Table 2.1. The payoff matrix for the Penny Flip game (the Picard's payoffs).

[1] Captain Picard and Q are characters from a television series — *Star Trek: The Next Generation*.

F and N denote *flip* and *no flip* respectively. The rows are associated with Picard's strategies and the columns with Q's. A strategy profile determines the game's result and so this is the value at the intersection of the selected row and column. The values 1 and -1 are the payoffs for Picard where 1 means a win and -1 otherwise. For example, if the Captain decides not to flip the penny (the N row) and Q decides to flip it in his first move and not to flip it in his second move (the F N column), the final state of the penny would be tails up, which means victory for Picard.

Considering his best course of action, Captain Picard instantly recognizes that there is no *Nash equilibrium* (see Def. 2.2.3), ie. a strategy profile such that no player would like to change his decision knowing the other player's choice, thus being a natural solution which players would be likely to agree upon. This is due to the simple fact that the game's outcome is determined by the parity of the overall number of flips — the game is won by Picard if the number of flips is odd (otherwise, it is won by Q). And as both players can always change the parity of this number by switching their strategy, for each strategy of each player there is a counter strategy that the other player can employ to secure his victory.

However, as Captain Picard remembered from his lectures, John Nash proved the following theorem:

Theorem 2.3.1 ([23])**.** *Every finite game has at least one probabilistic Nash equilibrium.*

Proof. Cf. [23] □

In this case, the probabilistic Nash equilibrium is such that Picard plays each of his two pure strategies with probability $\frac{1}{2}$ and Q each of his four pure strategies with probability $\frac{1}{4}$, namely the uniform distribution of the pure strategies for both players.

This probabilistic Nash equilibrium gives the expected payoff of 0 for both players, so Captain Picard knows that the outcome of the game can be no worse than him winning and losing roughly the same number of times, when he implements the mixed strategy of randomly flipping or not flipping with the same probability. Considering the gravity of the current situation Picard agrees to play this disturbingly simple game of Q's invention without further ado. Unfortunately, he loses... In this moment, a strange thing happened: Q made a proposition to replay the game and Captain Picard agreed believing that his chances are at least 50/50. However, he lost again. And again. And ten more times. What happened?

2.3.2. The quantum Penny Flip (as Q sees it)

Let us present the arrangement of the penny as a normalized vector from a two-dimensional complex Hilbert space $\mathcal{H}_2$ (ie. a qubit) with the basis $B = \{|0\rangle, |1\rangle\}$, where $|0\rangle$ and $|1\rangle$ denote the state of the penny being heads up and tails up respectively. Thus, at every moment of the game the state of the penny is described by the Equation 2.1.

The actions which players take in order to modify the penny's state are represented as unitary operators on $\mathcal{H}_2$. For example, Captain Picard can flip the penny, which corresponds to the X Pauli operator, or leave it as it is, which in turn corresponds to the identity operator I:

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad I \equiv \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

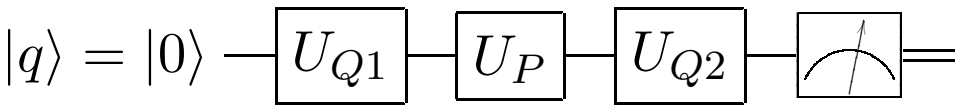

Figure 2.1. The quantum circuit diagram of the Penny Flip game.

Now we can introduce the quantum scheme of the Pennny Flip game. The game consists of five consecutive steps, as described in the beginning of this section (2.3), and each of these steps has a counterpart element in the quantum scheme (see Figure 2.1):

1. **The quantum state** is first prepared to be $|\mathbf{0}\rangle$.
2. **Q** applies his unitary operator U_{Q1} to **modify the quantum state**. The state is now $U_{Q1}|0\rangle$.
3. **Picard** applies his unitary operator U_P to **modify the quantum state**. The state is now $U_P U_{Q1}|0\rangle$.
4. **Q** applies his unitary operator U_{Q2} to **modify the quantum state**. The state is now $U_{Q2}U_P U_{Q1}|0\rangle$.
5. Finally, the quantum state is **measured**. If we express it as in the Equation 2.1, then it will collapse to the state $|0\rangle$ with probability $|\alpha|^2$ and to the state $|1\rangle$ with probability $|\beta|^2$. **Q wins** if the measured result is $|\mathbf{0}\rangle$. Otherwise, **Picard wins**.

The trick is that Picard only knows about classical strategies: no flip — I and classical flip — X and assumes the same about Q, so he does not recognize the danger of the Q's second action (it does not give him any advantage in the classical game as it was shown earlier). Q, however, does know that he can utilize *any* unitary operators U_{Q1} and U_{Q2}.

Suppose that Q chooses the Hadamard gate for both actions:

$$U_{Q1} = U_{Q2} = H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} . \tag{2.18}$$

Then after U_{Q1} the quantum state will be

$$U_{Q1}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) . \tag{2.19}$$

Regardless of what strategy is chosen by Picard the quantum state will remain as it was:

$$IU_{Q1}|0\rangle = XU_{Q1}|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = U_{Q1}|0\rangle \tag{2.20}$$

and finally after applying U_{Q2}:

$$U_{Q2}U_{Q1}|0\rangle = H\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right) = \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right) = |0\rangle , \tag{2.21}$$

thus Q can always win.

2.3.3. The "weak advantage" of the quantum player

Meyer also shows that in every two-person zero-sum game the quantum player has a "weak advantage" over the classical player, ie.:

Theorem 2.3.2 ([4])**.** *There is always a mixed/quantum equilibrium for a two-person zero-sum game, at which the expected payoff for the player utilizing a quantum strategy is at least as great as his expected payoff with an optimal mixed strategy.*

Proof. Cf. [4]. □

It comes from the fact that the effect of every mixed classical strategy on the quantum state can be reproduced by a quantum strategy (as it is shown in the proof of the aforementioned theorem), therefore the quantum player can simulate all mixed classical strategies and thus he is at least as "powerful" as the classical player.

2.4. Characteristics of a quantum game

Having introduced the first quantum game, we can now ask: what makes a game quantum? What are the characteristics of a quantum game? Looking closely at the Meyer's Penny Flip game we can see that it has these three properties:

1. **It uses concepts of quantum computing.**
   The state of the game is a quantum state; players act on this state by applying unitary operators; the quantum state is measured to obtain the game's result.

2. **It is reducible to its classical counterpart.**
   If the set of pure strategies is limited to the classical strategies: I — no flip and X — flip, then the game reduces to its classical version (see Subsection 2.3.1).

3. **It reveals properties beyond those which are found in its classical counterpart.**
   It has been shown that the quantum player can win the Penny Flip game regardless of the classical player's strategy, which is a major improvement on the mixed classical Nash equilibrium with the expected payoff of 0 for both players.

Henceforth it will be assumed that the term *quantum game* means a game that has the above characteristics.

2.5. The Prisoner's Dilemma

The next quantum game which will be discussed is the quantum Prisoner's Dilemma. The realization of this game will be the topic of the subsequent chapters. In this section a description of the classical

version of the Prisoner's Dilemma will be followed by its quantum formulation, namely the EWL scheme as it was introduced by Eisert, Wilkens, and Lewenstein [7].

2.5.1. The classical Prisoner's Dilemma

The Prisoner's Dilemma is a very well known two-person nonzero-sum game, which is interesting due to the fact that rational players choose a non-cooperative ("self-interested") strategy while a greater reward is offered when both players cooperate. The game can be interpreted as follows:

Alice and *Bob* are accused of jointly committing a crime and are being held separately. A clever prosecutor comes to each prisoner and makes them an offer (independently):

– If they both betray each other (*defect*), they will have to serve three years in prison.

– If they both remain silent (*cooperate*), they will serve one year in prison.

– If, however, only one of them defects and the other remains silent, the defector will be set free and the one who was silent will serve as many as five years in prison.

One can formalize the above story as a payoff matrix (see Table 2.2 below), where each payoff represents the number of prison years avoided from the heaviest punishment — five years' imprisonment.

Alice \ Bob	Cooperate	Defect
Cooperate	3 / 3	5 / 0
Defect	0 / 5	1 / 1

Table 2.2. The payoff matrix for the Prisoner's Dilemma (common payoff values).

The payoff values given in the above matrix are a common example [9]. Some sources give a more general form of the payoff matrix [24] (see Table 2.3 below).

Alice \ Bob	Cooperate	Defect
Cooperate	r / r	t / s
Defect	s / t	p / p

Table 2.3. The payoff matrix for the Prisoner's Dilemma (the general form), where $s < p < r < t$ and $2r > s + t$.

The *dominant strategy* for this game, ie. such a strategy that always leads to a payoff *no worse* than any other strategy, is *Defect* (it is also a *strictly* dominant strategy because it always leads to a payoff which is *greater* than any other strategy). For example, assuming Bob will *Cooperate*, Alice can go for the moderately good *r*eward for mutual cooperation r, but it is better to exploit Bob (by betraying him) to obtain the biggest reward t (*t*emptation). On the other hand, when Bob *Defect*s, cooperation will lead to the worst payoff s (the *s*ucker's payoff) and defection will secure the least worst *p*unishment payoff p.

Proposition 2.5.1. *Consider an n-player normal-form game $G = (N, S, p)$. If $s = (s_1, \ldots, s_n) \in S$ is a joint strategy such that each s_i ($i \in \{1, \ldots, n\}$) is strictly dominant, then s is a unique Nash equilibrium in G.*

Proof. Each s_i is a strictly dominant strategy, thus by the definition of strict dominance it has a greater payoff than all other strategies of player i for all possible joint strategies of the opponents. This implies that s_i is a best response for all possible joint strategies of the opponents. Thus s_i is also a best response for s_{-i}. As this is the case for each s_i, then s is a Nash equilibrium by its definition.

Now we have to prove that s is the only Nash equilibrium in G. We will do this by indirect proof. Suppose, that there is a Nash equilibrium s' other than s. Then each s'_i is a best response to s'_{-i}, thus s'_i has a maximum possible payoff when opponents play s'_{-i}. However, s_i has a greater payoff than all other strategies for any opponents' joint strategy. This leads to contradiction, thus s is a unique Nash equilibrium in G. □

Since the strictly dominant strategy for both players is to *Defect*, this implies that there is a Nash equilibrium, namely the *Defect-Defect* joint strategy, and it is the only equilibrium (by virtue of Proposition 2.5.1).

Then why is this game called the Prisoner's *Dilemma*? It comes from the fact that although there is only one stable solution — *Defect-Defect*, which gives reward p for both players, there is also another solution, namely *Cooperate-Cooperate*, which would give both players a greater reward r. In other words, the Nash equilibrium for this game is not *Pareto efficient*.

2.5.2. A quantum generalization of the Prisoner's Dilemma

Eisert, Wilkens, and Lewenstein proposed a quantum formulation of the Prisoner's Dilemma, which we will call the *EWL protocol* or the *EWL scheme* (see Figure 2.2 below). It is worth mentioning that this "scheme applies to any two-player binary choice symmetric game." ([7])

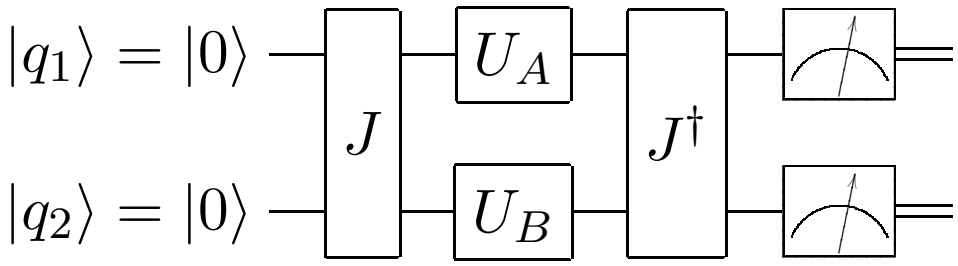

Figure 2.2. The quantum circuit diagram of the EWL scheme.

First, the quantum state is prepared as

$$|\psi_0\rangle = J|00\rangle \tag{2.22}$$

and J (for the maximally entangled form of the scheme, on which we will focus) is given by

$$J = \frac{1}{\sqrt{2}}(I + iX)\,. \tag{2.23}$$

The players' strategic decisions (*Cooperate* and *Defect*) are encoded on qubits and each qubit is mapped to a certain player's decision — for example the quantum states $|0\rangle$ and $|1\rangle$ may represent the decisions *Cooperate* and *Defect* respectively (as it will be assumed throughout this thesis). The players, traditionally named *Alice* and *Bob*, encode their strategies with unitary operators (quantum gates) U_A and U_B on their respective qubits. If the universal one-bit quantum gate is parameterized as

$$U(\theta,\phi,\lambda) = \begin{bmatrix} e^{-i(\phi+\lambda)/2}\cos\left(\frac{\theta}{2}\right) & -e^{-i(\phi-\lambda)/2}\sin\left(\frac{\theta}{2}\right) \\ e^{i(\phi-\lambda)/2}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)/2}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}, \quad \theta \in [0,\pi],\ \phi,\lambda \in [0,4\pi)\,, \tag{2.24}$$

then the quantum gates which encode the strategies *Cooperate* and *Defect* — C and D respectively — are given by

$$C \equiv U(0,0,0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad D \equiv U(\pi,0,0) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{2.25}$$

The final state is given by

$$|\psi_f\rangle = J^\dagger(U_A \otimes U_B)J|00\rangle \tag{2.26}$$

and this state is measured to obtain a pair of classical strategic decisions (*Cooperate* or *Defect*) — one for each player. The measurement will yield one of the following basis states: 00, 01, 10, 11. The payoff for a given player is determined by applying their payoff function (see Table 2.3) to the measurement result. As quantum computing is probabilistic in its nature, we will usually be interested in *expected* payoff values.

Eisert, Wilkens, and Lewenstein showed that if we consider a limited set of strategies which can be parameterized with only two parameters:

$$U'(\theta,\phi) = \begin{bmatrix} e^{i\phi}\cos\left(\frac{\theta}{2}\right) & \sin\left(\frac{\theta}{2}\right) \\ -\sin\left(\frac{\theta}{2}\right) & e^{i\phi}\cos\left(\frac{\theta}{2}\right) \end{bmatrix}, \quad \theta \in [0,\pi],\ \phi \in [0,\frac{\pi}{2}]\,, \tag{2.27}$$

(D, D) ceases to be a Nash equilibrium and there is a new equilibrium (Q, Q), where

$$Q \equiv U'\left(0, \frac{\pi}{2}\right) = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix} \tag{2.28}$$

and the expected payoff values are $\$_A(Q, Q) = \$_B(Q, Q) = r$ ($\$_A$ and $\$_B$ denote the payoff functions for Alice and Bob respectively). As the new Nash equilibrium is also Pareto efficient, it is said that "the players escape the dilemma." ([7]) However, as Benjamin and Hayden indicated,

> it seems unlikely that the restriction to the set [of strategies (see Eq. 2.27 as opposed to Eq. 2.24)] can reflect any reasonable physical constraint (limited experimental resources, say) because this set is not closed under composition. [25]

In fact, in the space of all quantum strategies (see Eq. 2.24) there is no equilibrium as for every strategy $U(\theta, \phi, \lambda)$ there is a counter strategy $U(\theta + \pi, -\phi - \frac{\pi}{2}, \lambda - \frac{\pi}{2})$ which gives the countering player the optimal outcome t [25]. However, if we restrict the set of strategies to $\{U(\theta, 0, 0) \mid \theta \in [0, \pi]\}$ the game reduces to its mixed classical version, in which players choose randomly between the classical strategies *Cooperate* and *Defect* and the parameter θ determines the probability distribution ($\theta = 0$ means that a player will always *Cooperate* and $\theta = \pi$ means that they will always *Defect*).

We will now show that the EWL scheme is indeed a quantum game in our sense (see Section 2.4). Clearly, it uses concepts of quantum computing: players utilize their strategies by applying unitary operators on the prepared quantum state, the state is measured to obtain the final outcome. It is also reducible to its classical counterpart as we have shown above. Lastly, it shows some new intriguing properties. Despite the lack of Nash equilibrium in the most general case, it turns out that there is one, if players restrict themselves to a certain subset of strategies, which will be shown below.

As we said before, for each strategy $U(\theta, \phi, \lambda)$ there is a counter strategy $U(\theta + \pi, -\phi - \frac{\pi}{2}, \lambda - \frac{\pi}{2})$. Suppose Alice chooses strategy $A \equiv U(\theta, \phi, \lambda)$. Then the best response for Bob would be to play $B \equiv U(\theta + \pi, -\phi - \frac{\pi}{2}, \lambda - \frac{\pi}{2})$. If, however, Alice knew that Bob would play B, she could play $A' \equiv U(\theta + 2\pi, \phi, \lambda - \frac{\pi}{2})$, as it is the best response for B. Consequently, the best response for A' is $B' \equiv U(\theta + 3\pi, -\phi - \frac{\pi}{2}, \lambda - \frac{3\pi}{2})$ and then it turns out that the countering strategies form a cycle, as the best response for B' is A. If players mix the above strategies, ie. Alice plays a mixed strategy $\cos^2 \frac{\gamma_A}{2} A + \sin^2 \frac{\gamma_A}{2} A'$, $\gamma_A \in [0, \pi]$ and Bob plays $\cos^2 \frac{\gamma_B}{2} B + \sin^2 \frac{\gamma_B}{2} B'$, $\gamma_B \in [0, \pi]$, then there is one Nash equilibrium, namely $\gamma_A = \gamma_B = \frac{\pi}{2}$, for which the payoffs are $\$_A = \$_B = \frac{s+t}{2}$ [26]. In fact, if any player chooses $\gamma = \frac{\pi}{2}$, the payoffs for both players will be fixed on $\frac{s+t}{2}$.

This shows that there are infinitely many restricted games and for every one of them there is a Nash equilibrium giving players an equal payoff, which is only slightly worse than the payoff for mutual cooperation (note that one of the constraints for the Prisoner's Dilemma is $r > \frac{s+t}{2}$).

2.6. Summary

We have introduced basic concepts of quantum computation and game theory, explained the concept of a quantum game and discussed the Prisoner's Dilemma — its classical version as well as the quantum generalization introduced by Eisert, Wilkens, and Lewenstein.

3. Quantum circuits on IBM Q

In this chapter we introduce IBM Q and issues concerning the realization of quantum algorithms on the IBM Q quantum computers. We will give some insight into the physical realization of the IBM Q computers and then proceed to cover the topics of quantum errors and quantum error correction methods. This chapter ends with a short section on circuit optimization.

3.1. An introduction to IBM Q

As the IBM Q's homepage states: "IBM Q is an industry first initiative to build universal quantum computers for business, engineering and science." ([18]) It offers a number of quantum computing devices and simulators available publicly on the cloud, which can be accessed through the *IBM Q Experience* platform [2] or the Web API.

The first quantum processor of the IBM Q project, which implemented 5 qubits, was opened to the public in 2016 [1]. Since then the project is under constant development offering new devices with more and more qubits as well as new tools, such as *Qiskit* [3] — a Python library for quantum computation on IBM Q. At the time of writing there are four active public IBM Q quantum devices: *IBM Q Melbourne* (14 qubits), *IBM Q Yorktown* (5 qubits), *IBM Q Vigo* (5 qubits), and *IBM Q Ourense* (5 qubits) as well as one premium device: *IBM Q Tokyo* (20 qubits) available for IBM Q clients only [18].

3.2. Physical realization

The key element of any IBM Q quantum device is the *transmon* qubit [18]. The name is an abbreviation of the term *transmission line shunted plasma oscillation* qubit, which is a type of superconducting qubit [27]. The transmon qubits are connected with each other by a superconducting bus resonator [28] to form a network (for an example of such network see Figure 3.1 below). The coupling of qubits determines the possible combinations of controlled-NOT quantum gates. As "[t]he IBM Q experience uses the cross-resonance interaction as the basis for the CX-gate[] [and] [t]his interaction is stronger when choosing the qubit with higher frequency to be the control qubit, and the lower frequency qubit to be the target, [] the frequencies of the qubits determines the direction of the gate." ([28]) The gate directions for IBM Q Tenerife are shown in Figure 3.2 below. This whole device is kept in a very low temperature (around 15 mK [18]) in order to mitigate the effects of *quantum decoherence* (see Section 3.3 below).

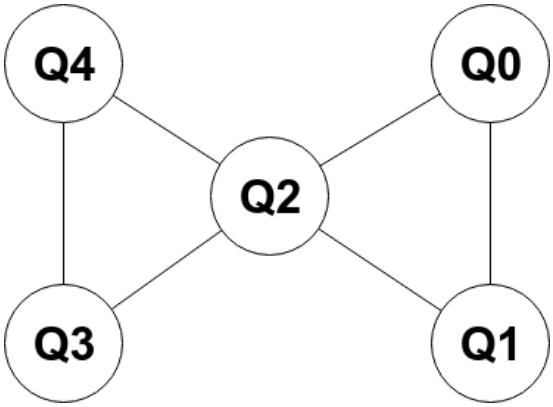

Figure 3.1. The qubit coupling map of IBM Q Tenerife.

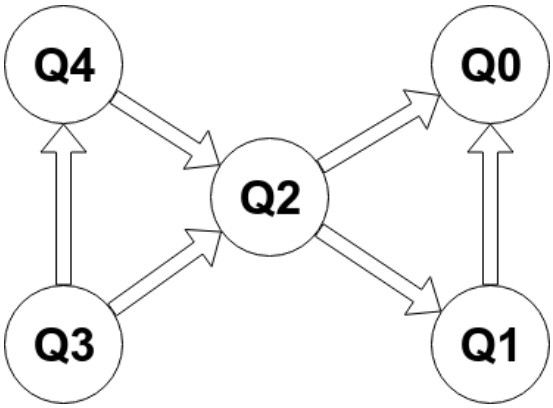

Figure 3.2. The controlled-NOT gate directions in IBM Q Tenerife (as for 30th July 2019).

The IBM Q devices are capable of applying any one-qubit quantum gate as well as the controlled-NOT gate (in the possible directions, see Figure 3.2) [28], which is sufficient for implementation of any multi-qubit quantum gate [19].

3.3. Quantum errors

There are two types of errors in quantum devices due to their source: *coherence errors* (or *retention errors*) and *gate errors* (or *operational errors*) [29].

3.3.1. Coherence errors

Coherence errors occur as a consequence of the fact that quantum states have a tendency to decohere with time due to their interactions with the environment. It is therefore essential to limit those interactions as far as possible, for example by lowering the temperature of the device to an extreme point and thus taking away almost all of the particles' kinetic energy (virtually "freezing" them in place). Obviously, it

is not desirable to isolate a quantum state completely, as this would render it useless for computational purposes, which require the ability to apply quantum operations and perform measurements.

Ensuring high qubit quality is one of the main challenges of the current quantum computing technology. There are two metrics which quantify the ability of a quantum device to retain a quantum state. The *T1 Coherence Time* or *Relaxation Time* is a time constant associated with the relaxation of a qubit, ie. its decay from the high-energy state $|1\rangle$ to the low-energy state $|0\rangle$, whereas the *T2 Coherence Time* or *Dephasing Time* is associated with the dephasing of a qubit, ie. the randomization of the qubit's phase.

For example, the average values of T1 and T2 for IBM Q Tenerife are about $44\ \mu s$ and $25\ \mu s$ respectively (as for 31st May 2019) [30].

3.3.2. Gate errors

Quantum operations in a quantum device are not perfect, which results in errors called the gate errors (or operational errors). The key concept in quantifying the gate errors is the *fidelity* [10] of two given quantum states, which can be understood as a measure of the "closeness" of those states. If we have two quantum states represented as density matrices ρ and σ, the fidelity $F(\rho, \sigma)$ of those states is defined as

$$F(\rho, \sigma) = \left[\mathrm{Tr} \sqrt{\sqrt{\rho}\sigma\sqrt{\rho}} \right]^2 . \tag{3.1}$$

We can then characterize the quantum gate by its average fidelity, ie. the average fidelity of the output state, ideal over experimental. Alternatively, we can give operational error-rates defined as $1 - F$, where F is the average fidelity [31].

In the case of IBM Q Tenerife the average idelities of 1-qubit gates and 2-qubit gates are about 99.89% and 95.54% respectively and the average readout fidelity is about 84.7% (as for 31st May 2019) [30].

3.4. Quantum error correction

As it was mentioned in the section 3.3 concerning quantum errors, the results obtained on current quantum devices are far from ideal. There are, however, some methods of mitigating those errors and these are grouped together under the name of *quantum error correction*. It should be mentioned that the possibility of employing quantum error correction on IBM Q devices is limited, as they do not allow quantum operations after measurement.

In contrast to classical bits, there are a number of kinds of errors that can occur in quantum devices [10]. We will be mostly concerned with one specific kind — the *bit flip*. The bit flip error results in a change of a qubit state from $|0\rangle$ to $|1\rangle$ and vice versa. The effect of this error is the same as applying the Pauli operator X with a certain probability. This is the only kind of quantum errors which has a counterpart in classical computation — when a classical bit flip occurs, a bit flips its state from 0 to 1 and vice versa.

There are also other kinds of quantum errors, which do not have classical counterparts, such as the *phase flip*, which has the same effect as applying the Pauli operator Z with a certain probability. As the *phase flip* is not directly measurable with measurements in the computational basis (it can be, however, measured in other basis), it will not be a concern for this thesis.

In this thesis two methods of quantum error correction were used. The first one is based on the idea of repetition codes and the second one uses the knowledge of the measurement error distribution to mitigate this kind of error.

3.4.1. The repetition code

The idea of repetition codes is widely used in classical error correction [32]. Suppose that we store a single bit of information and an error, which can only be the bit flip in this case, occurs with probability $p < \frac{1}{2}$. Thus, when we read the value of this bit we will get the erroneous result with probability p. We can improve our situation at the cost of memory usage if we *encode* the *logical* state of the bit onto a number of *physical* bits. That is for example if we repeat the bit $d = 5$ times:

$$0 \to 00000\,, \quad 1 \to 11111\,.$$

When we want to read the *logical* value of the bit, we need to read the encoded state and *decode* it simply by choosing the value that occurs most often (this is called *majority voting*). The probability of a *logical error*, ie. decoding a different value that the one which was encoded, is equal to the probability that a majority of bits will flip (we can eliminate the risk of a tied vote, when we set d to be an odd number) and that is equal to $p^{\lceil \frac{d}{2} \rceil}$.

We can use a similar method for quantum error correction [33]. Let us consider the simplest case of a single-qubit state. First, we prepare the quantum state on d qubits. We also have to prepare one ancilla qubit for each pair of state qubits. Then, we apply two CNOT gates on each ancilla qubit controlled by their respective state qubits (see Figure 3.3). Finally, all qubits are measured to obtain the final output of the quantum correction code $R \in \{0,1\}^{2d-1}$.

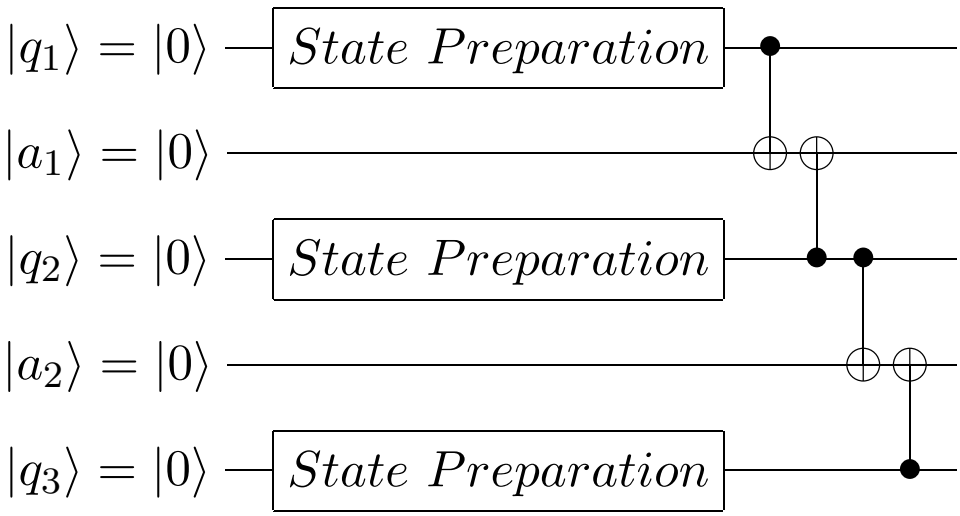

Figure 3.3. A general repetition quantum correction circuit for a single-qubit state and repetition number $d = 3$.

In order to decode the logical bit value we need to know probabilities $\pi(R \mid E)$ of obtaining measurement result R, when the encoded value was $E \in \{0, 1\}$. This probability can be determined experimentally by repetitive measurements of the quantum correction code for each $E \in \{0, 1\}$. Then we can calculate a lookup table for decoding logical states $l : \{0, 1\}^{2d-1} \to \{0, 1\}$ as shown in the Algorithm 1 listing below.

Input: Measurement distributions $\pi_E : \{0, 1\}^{2d-1} \to [0, 1]$ for each basis state $E \in \{0, 1\}$
Output: A lookup table $l : \{0, 1\}^{2d-1} \to \{0, 1\}$
for $s \in \{0, 1\}^{2d-1}$ **do**
  $candidates \leftarrow \{0, 1\}$;
  $maxp \leftarrow 0$;
  for $state \in \{0, 1\}$ **do**
    if $s \in D_{\pi_E}$ **then**
      if $\pi_s(state) > maxp$ **then**
        $maxp \leftarrow \pi_s(state)$;
        $candidates \leftarrow \{state\}$;
      end
      else if $\pi_s(state) = maxp$ **then**
        $candidates = candidates \cup \{state\}$;
      end
    end
  end
  $l(s) \leftarrow random(candidates)$;
end

Algorithm 1: Calculating lookup table l from experimental data.

Having a measurement distribution π we can determine the corrected distribution π' as shown in the Algorithm 2 listing below.

Input: A measurement distribution $\pi : \{0,1\}^{2d-1} \to [0,1]$, a lookup table
$l : \{0,1\}^{2d-1} \to \{0,1\}$

Output: The corrected distribution $\pi' : \{0,1\} \to [0,1]$

Initialize π' to 0 for all domain elements;

for $s \in D_\pi$ **do**

  $\pi'(l(s)) \leftarrow \pi'(l(s)) + \pi(s)$;

end

Algorithm 2: Determining the corrected distribution π' based on a measurement distribution π and a lookup table l.

3.4.2. Measurement error mitigation

The second method of quantum error correction used in this thesis is based on the idea that there may be a bias towards certain measurement results. We can experimentally determine this bias and then apply a correction to the measurement results. This is called measurement error mitigation and is performed as follows.

First a series of calibration circuits for a chosen number of qubits n is created, each of which consists of two parts: preparation of one of the basis states and measurement. A sample set of calibration circuits for two qubits is shown in Figure 3.4 below.

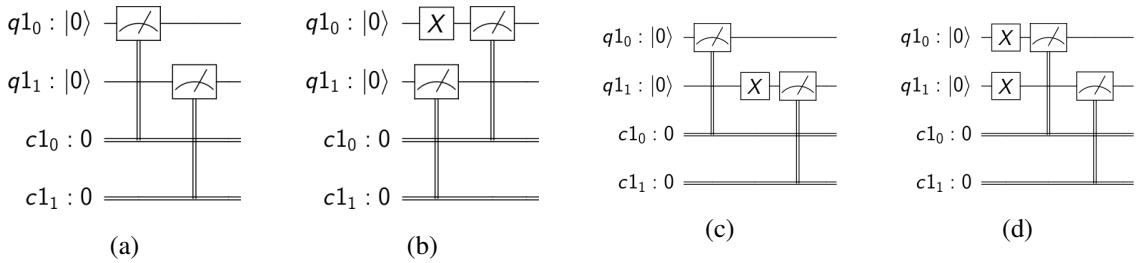

Figure 3.4. Calibration circuits for two qubits. Quantum states to be measured are: (a) $|00\rangle$, (b) $|10\rangle$, (c) $|01\rangle$, (d) $|11\rangle$.

Based on the measurement results of these circuits the calibration matrix $A = (a_{ij})_{i,j\in\{1,\ldots,2^n\}}$ is then calculated. Each a_{ij} corresponds to the conditional probability of obtaining measurement result i, if the prepared state was j. Thus we can treat A as a stochastic matrix, which transforms vectors of probability. We assume that if we prepare a quantum state to yield measurement results according to some vector x, we will get measurement results according to a vector b and

$$Ax = b\,. \tag{3.2}$$

Knowing the calibration matrix A and the experimental measurement probabilities b, we can solve the system of linear equations Eq. 3.2 to obtain the corrected measurement distribution x.

3.5. Circuit optimization

Apart from quantum error correction methods there is another way to increase the fidelity of quantum operations, namely circuit optimization, which involves performing some preprocessing on the classical computer in order to decrease the size of the quantum circuit, especially the number of the CNOT gates.

In Qiskit there is a module called the *transpiler*, which is responsible for analysis and transformation of user-defined circuits and circuit optimization is a part of its functionality. The processing of the transpiler module is defined in terms of *transpiler passes*. These are circuit processing algorithms which can be chained together to form a pipeline. A user can define, which transpiler passes should be executed and in which order. There are, however, four pre-defined transpiler pipelines which correspond to four levels of optimization:

- Level 0 — the transpiler does no explicit optimization. It only maps the virtual qubits defined in the circuit to the physical qubits on the backend.
- Level 1 — the transpiler does everything from the previous level and also does some minor optimization by collapsing adjacent quantum gates.
- Level 2 — the transpiler uses a noise-adaptive mapping [34] to map the virtual qubits to the physical qubits and then performs gate cancellation using commutativity rules.
- Level 3 — the transpiler does everything from the previous level and also does resynthesis of two-qubit unitary blocks [35].

It should also be noted that there is a circuit element in Qiskit, namely the *barrier*, which allows to divide a circuit into parts which will be optimized independently.

3.6. A note on changes in IBM Q during writing of this thesis

As the IBM Q project is under constant development, new features were being added to the Qiskit Python library and even new quantum devices (IBM Q Vigo and Ourense) were made available during writing of this thesis. Moreover, the Qiskit API was being changed from time to time, which made the work on this thesis more difficult.

3.7. Summary

We have introduced IBM Q and concepts related to realization of quantum algorithms on real quantum devices, especially the IBM Q quantum computers. After this and the previous chapter about quantum games we are ready to present the solution of this thesis — the realization of the EWL scheme on an IBM Q device — in the next chapter.

4. Solution

This chapter presents the EWL scheme realization on an IBM quantum device. It starts with the methodology and a detailed description of the method used to decompose matrices and then proceeds to presents the results of each step taken to create an optimized quantum circuit for execution on IBM Q. The chapter ends with a description of quantum error correction method application.

4.1. Methodology

The research presented in this thesis was carried out as follows. The first milestone was realization of the EWL scheme on an IBM Q quantum computer. In order to accomplish this goal it was necessary to find the decomposition of the J gate into a sequence of fully controlled single-qubit gates, which was done using the method introduced by Li, Roberts, and Yin [36] as it is described in Section 4.2 below. The decomposed EWL scheme was tested using a classical simulator available in Qiskit [3]. When the validity of the EWL Circuit had been ascertained, executions on real IBM Q quantum devices were performed. As the results obtained on quantum devices were unsatisfactory due to quantum decoherence, attempts were made to mitigate their influence on the results. Two methods of quantum error correction were applied independently (see Section 4.3 below).

4.2. EWL scheme realization

As IBM Q devices implement only single-qubit quantum gates and their controlled counterparts, it is necessary to represent the entangling gate J (see Figure 2.2 and Eq. 2.23) as a combination of the former in order to realize the EWL scheme on an IBM Q device. In fact, it can be shown that:

Theorem 4.2.1 ([36])**.** *Every quantum gate acting on n-qubit registers can be expressed as no more than* $2^{n-1}(2^n-1)$ *fully controlled single-qubit gates chosen from* 2^n-1 *classes, where the quantum gates in each class share the same* $n-1$ *control qubits.*

Proof. Cf. [36]. □

4.2.1. Method description

Li, Roberts, and Yin introduced a method of decomposing unitary matrices [36], which can be used to decompose arbitrary quantum gates into fully controlled single-qubit gates.

Definition 4.2.1 (P-unitary matrix [36])**.** Let $P = (j_1, j_2, \ldots, j_d)$ be such that the entries of P correspond to a permutation of $(1, 2, \ldots, d)$. A two-level unitary matrix is called a **P-unitary matrix of type** k for $k \in \{1, 2, \ldots, d-1\}$ if it is obtained from I_d by changing a principal submatrix with row and column indexes j_k and j_{k+1}.

For example if $P = (j_1, j_2, j_3, j_4) = (1, 2, 4, 3)$, then the P-unitary matrices of type 1, 2 and 3 have the forms

$$\begin{bmatrix} * & * & 0 & 0 \\ * & * & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & * & 0 & * \\ 0 & 0 & 1 & 0 \\ 0 & * & 0 & * \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{bmatrix},$$

respectively.

Proposition 4.2.1 ([36])**.** *Every $d \times d$ unitary matrix U can be written as a product of no more than $d(d-1)/2$ P-unitary matrices. Moreover, these P-unitary matrices can be chosen to have any determinants with modulus 1 as long as their product equals $\det(U)$.*

Proof. Cf. [36]. □

We will illustrate this proposition with an example from the work of Li, Roberts, and Yin [36]. Let $d = 4$, $P = (j_1, j_2, j_3, j_4) = (1, 2, 4, 3)$, and

$$U = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}. \tag{4.1}$$

Let $\mu_1, \mu_2, \ldots, \mu_6$ be such that $\mu_1, \mu_2, \ldots, \mu_6 \in \{z : |z| = 1\}$ and $\mu_1 \mu_2 \ldots \mu_6 = \det(U)$.

First we consider the column of U labeled by the first entry of P: $j_1 = 1$. We choose P-unitary matrix U_1 of type 3 such that $\det(U_1) = \bar{\mu}_1$ and the $(j_4, j_1) = (3, 1)$ entry of $U_1 U$ is 0 as follows. Let $u_1 = \sqrt{|a_{31}|^2 + |a_{41}|^2}$ and

$$U_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{\bar{\mu}_1 a_{41}}{u_1} & \frac{-\bar{\mu}_1 a_{31}}{u_1} \\ 0 & 0 & \frac{\bar{a}_{31}}{u_1} & \frac{\bar{a}_{41}}{u_1} \end{bmatrix}. \quad \text{Then} \quad U_1 U = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ u_1 & a'_{42} & a'_{43} & a'_{44} \end{bmatrix}. \tag{4.2}$$

Next we choose P-unitary matrix U_2 of type 2 such that $\det(U_2) = \bar{\mu}_2$ and the $(j_3, j_1) = (4, 1)$ entry of U_2U_1U is 0 as follows. Let $u_2 = \sqrt{|a_{21}|^2 + u_1^2}$ and

$$U_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\bar{a}_{21}}{u_2} & 0 & \frac{u_1}{u_2} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{-\bar{\mu}_2 u_1}{u_2} & 0 & \frac{\bar{\mu}_2 a_{21}}{u_2} \end{bmatrix} . \quad \text{Then} \quad U_2U_1U = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ u_2 & a'_{22} & a'_{23} & a'_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a''_{42} & a''_{43} & a''_{44} \end{bmatrix} . \tag{4.3}$$

Now, we choose P-unitary matrix U_3 of type 1 such that $\det(U_3) = \bar{\mu}_3$, the $(j_2, j_1) = (2, 1)$ entry of $U_3U_2U_1U$ is 0 and the $(1, 1)$ entry of $U_3U_2U_1U$ is 1 as follows. Let

$$U_3 = \begin{bmatrix} \bar{a}_{11} & u_2 & 0 & 0 \\ -\bar{\mu}_3 u_2 & \bar{\mu}_3 a_{11} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \quad \text{Then} \quad V = U_3U_2U_1U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & a''_{22} & a''_{23} & a''_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a''_{42} & a''_{43} & a''_{44} \end{bmatrix} . \tag{4.4}$$

The first row of V is $(1, 0, 0, 0)$, because V is unitary.

Next we turn to columns of V labeled by $j_2 = 2$ and $j_3 = 4$. We can choose P-unitary matrices U_4 and U_5 of types 3 and 2, respectively, such that $\det(U_4) = \bar{\mu}_4$, $\det(U_5) = \bar{\mu}_5$, the $(j_4, j_2) = (3, 2)$ entry of U_4V is 0, and the $(j_3, j_2) = (4, 2)$ entry of U_5U_4V is also 0. Then we can choose P-unitary matrix U_6 of type 3 such that $\det(U_6) = \bar{\mu}_6$ and the $(j_4, j_3) = (3, 4)$ entry of $U_6U_5U_4V$ is 0. The intermediate matrices will have the following zero patterns:

$$U_4V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & * & * & * \end{bmatrix} , \quad \text{and} \quad U_5U_4V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{bmatrix} .$$

And finally

$$U_6U_5U_4V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = I_4 . \tag{4.5}$$

The $(j_4, j_4) = (3, 3)$ entry of $U_6U_5U_4V$ is 1 because

$$\det(U_6U_5U_4U_3U_2U_1) \det(U) = \bar{\mu}_6\bar{\mu}_5 \dots \bar{\mu}_1 \det(U) = |\det(U)|^2 = 1 . \tag{4.6}$$

In view of the above we can write

$$U = U_1^\dagger U_2^\dagger U_3^\dagger U_4^\dagger U_5^\dagger U_6^\dagger . \tag{4.7}$$

It is evident that the matrices $U_j^\dagger$ $(j = 1, \dots, 6)$ are also P-unitary matrices of the same type as their respective U_j matrices. It should also be mentioned that some of the U_j may be equal to I_d (ie. may be skipped) if the entry to be eliminated is already 0.

We are just a step away from finding the decomposition of quantum gates (which are equivalent to unitary matrices in a given basis) into *fully controlled single-qubit gates*. This last step is to find the subset of P-unitary matrices which can be identified with the latter. Given a system with n qubits, we will denote a fully controlled gate as $C^{n-1}V$, where $V = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}$ is a single-qubit gate. $C^{n-1}V$ has $n-1$ control qubits valued 0 or 1, which determine the controlling subspace. For example, in the case of 4×4 unitary matrices there are four possible controlled gates. If we take the first qubit as the control qubit:

$$\begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} v_{11} & v_{12} & 0 & 0 \\ v_{21} & v_{22} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}, \quad \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & v_{11} & v_{12} \\ 0 & 0 & v_{21} & v_{22} \end{bmatrix} \end{array}.$$

If we take the second qubit as the control qubit:

$$\begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} v_{11} & 0 & v_{12} & 0 \\ 0 & 1 & 0 & 0 \\ v_{21} & 0 & v_{22} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}, \quad \begin{array}{c} \\ 00 \\ 01 \\ 10 \\ 11 \end{array}\!\!\begin{array}{c} \begin{array}{cccc} 00 & 01 & 10 & 11 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & v_{11} & 0 & v_{12} \\ 0 & 0 & 1 & 0 \\ 0 & v_{21} & 0 & v_{22} \end{bmatrix} \end{array}.$$

As Li, Roberts, and Yin noticed:

> In general, if we label the rows and columns of quantum gates (unitary matrices) acting on n qubits by binary sequences $x_1, \ldots, x_n$, then a $C^{n-1}V$ gate corresponds to a two-level matrix obtained from I_{2^n} by replacing its 2×2 principal submatrix lying in rows and columns $X = x_1 \ldots x_n$ and $\tilde{X} = \tilde{x}_1 \ldots \tilde{x}_n$ by V for two binary sequence X and $\tilde{X}$ [which] differ exactly in one of their terms, say, $x_i \neq \tilde{x}_i$. [36]

Definition 4.2.2 (Gray code [36])**.** Let n be a positive integer and $N = 2^n$. A **Gray code** G_n is an N-tuple $G_n = (X_1, \ldots, X_N)$ such that:

1. $X_1, \ldots, X_N$ are binary string representations of numbers $0, 1, \ldots, N-1$ arranged in a certain order.
2. Two adjacent strings X_j and X_{j+1} differ in only one position.
3. X_1 and X_N differ in only one position.

For example, $G_2 = (00, 01, 11, 10)$. If we substitute G_n for P in the definition of P-unitary matrix (see Def. 4.2.1), we will obtain *G_n-unitary matrices*, which can be identified with fully controlled single-qubit gates [36]. We can then apply G_n-unitary matrices to Proposition 4.2.1 to find the decomposition of any unitary matrix (ie. an arbitrary quantum gate) into fully controlled single-qubit gates.

4.2.2. Method application

In order to decompose the J gate the following steps were taken:

1. A Matlab implementation of the above scheme — `pud.m` [37] — was used to find the decomposition in the form of a sequence of unitary matrices.

2. The unitary matrices were expressed using the parameterization required by IBM Q to create a quantum circuit.

3. The quantum circuit was optimized in order to minimize the error on a quantum device.

The results of these steps are described in detail below.

The solution found by Matlab implementation is the following decomposition:

$$J = J_6 J_5 J_4 J_3 J_2 J_1 \,, \tag{4.8}$$

where

$$J_1 = J_6 \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -i & 0 \\ 0 & 0 & 0 & i \end{bmatrix} ,$$

$$J_2 = J_5 \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix} ,$$

$$J_3 \equiv \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} ,$$

$$J_4 \equiv \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ,$$

and thus

$$J^\dagger = J_1^\dagger J_2^\dagger J_3^\dagger J_4^\dagger J_5^\dagger J_6^\dagger \,. \tag{4.9}$$

The full EWL scheme realized using the above elementary gates is shown in Figure 4.1 below (*EWL Circuit*). The gate notation used in the Figure 4.1 is as follows:

$$A \equiv \begin{bmatrix} -i & 0 \\ 0 & i \end{bmatrix}, \qquad B \equiv \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix},$$
$$C \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, \qquad D \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix},$$

X denotes the NOT (or Pauli σ_x) gate, and U_A, U_B denote the gates with which Alice and Bob implement their respective strategies.

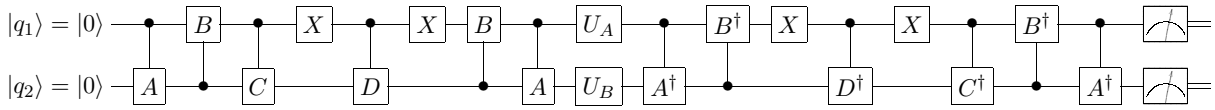

Figure 4.1. The quantum circuit diagram of the EWL Circuit. The gate notation is explained in the containing section.

In Qiskit single-qubit gates are given by the three parameters for the universal quantum gate $U(\theta, \phi, \lambda)$ (see Eq. 2.24), therefore it is necessary to find those parameters for our elementary gates. After some simple calculations we see that

$$A \equiv U\left(0, \frac{\pi}{2}, \frac{\pi}{2}\right), \qquad A^\dagger \equiv U\left(0, \frac{3\pi}{2}, \frac{3\pi}{2}\right),$$
$$B \equiv U\left(\pi, 2\pi, 0\right), \qquad B^\dagger \equiv U\left(\pi, 0, 0\right),$$
$$C \equiv U\left(\frac{\pi}{2}, 2\pi, 0\right), \qquad C^\dagger \equiv U\left(\frac{\pi}{2}, \pi, \pi\right),$$
$$D \equiv U\left(\frac{\pi}{2}, 3\pi, \pi\right), \qquad D^\dagger \equiv U\left(\frac{\pi}{2}, 0, 0\right).$$

As for U_A and U_B we can see that each player strategy is determined by three parameters $(\theta_i, \phi_i, \lambda_i) \in S_i = [0, \pi] \times [0, 4\pi)^2$, where i denotes Alice or Bob. Clearly, the game is infinite, as the set of strategy profiles $S = S_A \times S_B$ is continuously infinite.

Finally, the circuit was optimized for execution. We used preprocessing optimization (see Section 3.5) to optimize J and $J^\dagger$ gates which are player-independent parts of the EWL scheme. To achieve that, we put barriers to separate the player-dependent and player-independent parts of the circuit. The result is shown in Figure 4.2 below.

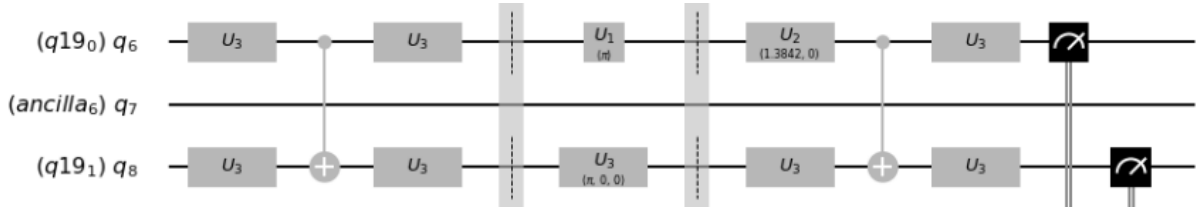

Figure 4.2. The quantum circuit diagram of the optimized version of the EWL scheme.

4.3. Application of quantum error correction methods

One of the objectives of this thesis was to apply quantum error correction methods and evaluate their influence on the results of the EWL scheme execution on an IBM Q device. Two of such methods were used: the repetition code and measurement error mitigation. They are described in detail in Section 3.4.

4.3.1. The repetition code

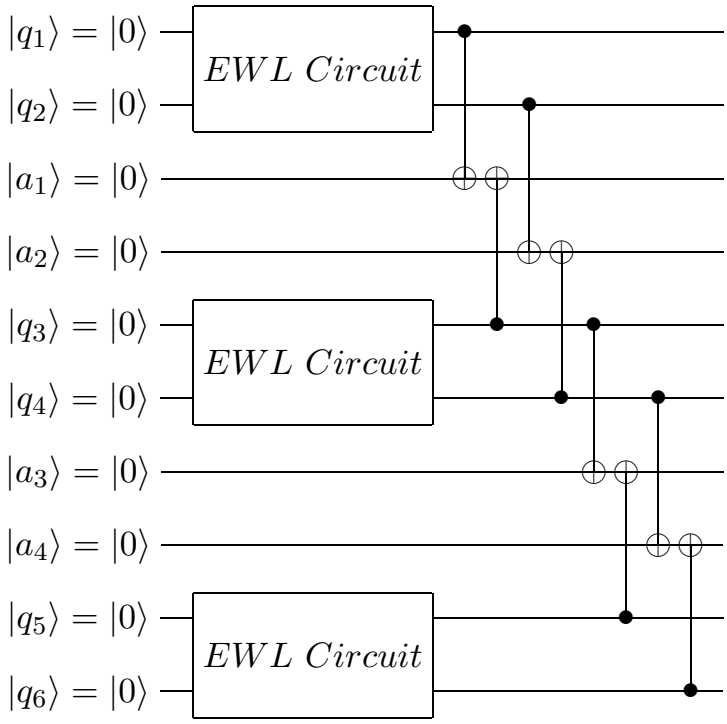

Figure 4.3. The repetition code circuit for the EWL scheme ($d = 3$).

The repetition code correction method is applied as follows:

1. The repetition code circuit is prepared for $d = 3$ (see Figure 4.3).
2. The repetition code circuit is executed on IBM Q for the set of strategies $S' = \{((\theta_A, 0, 0), (\theta_B, 0, 0)) \mid \theta_A, \theta_B \in \{0, \pi\}\}$, which produce the two qubit basis states $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ on the output of the EWL Circuit.
3. The lookup table l is prepared based on the results from the previous step (see Algorithm 1).
4. The repetition code circuit is executed on IBM Q for a given set of strategies.
5. The results are corrected using the lookup table l as described in Algorithm 2.

The algorithms were implemented in Python and quantum circuits were executed on IBM Q quantum devices using Qiskit API.

4.3.2. Measurement error mitigation

Measurement error mitigation is performed as it was described in Section 3.4. It is implemented by the Qiskit Python library (package `qiskit.ignis.mitigation.measurement`). The method is applied in the following steps:

1. The two-qubit calibration circuits for the EWL scheme are prepared (see Figure 3.4) by the `complete_meas_cal` function.
2. The calibration circuits are executed on IBM Q.
3. The calibration matrix is calculated based on the results of the calibration circuits as described in Section 3.4 by a fitter object — `CompleteMeasFitter`.
4. The EWL Circuit is executed on IBM Q for a given set of strategies.
5. Correction of the results from the previous step is performed based on the calibration matrix by solving the system of linear equations (3.2) by a filter object — `MeasurementFilter`.

This method was performed in Python using the Qiskit implementation and quantum circuits were executed using the Qiskit API.

4.4. Summary

We have presented the realization of the EWL scheme on an IBM Q quantum device along with two methods of quantum error correction. In the next chapter we will show the evaluation results of the above solution.

5. Evaluation

This chapter presents the evaluation of the solution given in the previous chapter. It starts by evaluating the EWL Circuit on a classical simulator without quantum errors to assess the validity of the circuit itself. Then it attempts to measure the influence of quantum errors of a real quantum device on the game results and the efficiency of the quantum error correction methods used in this thesis.

5.1. Perfect simulation

As described in Section 1.4 the first objective of this thesis was to find a form of the EWL scheme compatible with the IBM Q requirements. One can easily prove validity of the obtained circuit (see Figure 4.1) by recomposing the unitary matrices. We compared expected payoff values obtained theoretically and in the IBM Q simulator. These values are determined as follows. The measurement results of the EWL Circuit are mapped to payoff values according to Table 5.1 below (cf. Table 2.2).

measured result	p_A	p_B
00	3	3
01	0	5
10	5	0
11	1	1

Table 5.1. The mapping of measured outcomes to players' payoffs.

The expected payoff values are calculated based on the probability distribution of the measurement results. If $\pi : \{00, 01, 10, 11\} \to [0, 1]$ is such a distribution and $p_i : \{00, 01, 10, 11\} \to \mathbb{R}$ is the mapping shown in Table 5.1 for player i, then the expected payoff value for that player is given by

$$E(p_i) = \sum_{r \in \{00,01,10,11\}} \pi(r) \cdot p_i(r) \,. \tag{5.1}$$

In figure 5.1 below we show the payoff values for the first player which result from a chosen subset of joint quantum strategies. Strategies in the diagram are parameterized by function $f : [0, 1] \to S_p$,

where S_p is the set of strategies available to each player, as follows:

$$f(s) := \begin{cases} U(\pi - 2s\pi, 0, 0) & s \in [0, \frac{1}{2}] \\ U(0, (s - \frac{1}{2})\pi, (s - \frac{1}{2})\pi) & s \in (\frac{1}{2}, 1] \end{cases}, \tag{5.2}$$

where $U(\theta, \phi, \lambda)$ is a matrix parameterization given by Eq. 2.24. Thus, $f(0) = U(\pi, 0, 0) = X$, $f(\frac{1}{2}) = U(0, 0, 0) = I$, and $f(1) = U(0, \frac{\pi}{2}, \frac{\pi}{2}) = Z$.

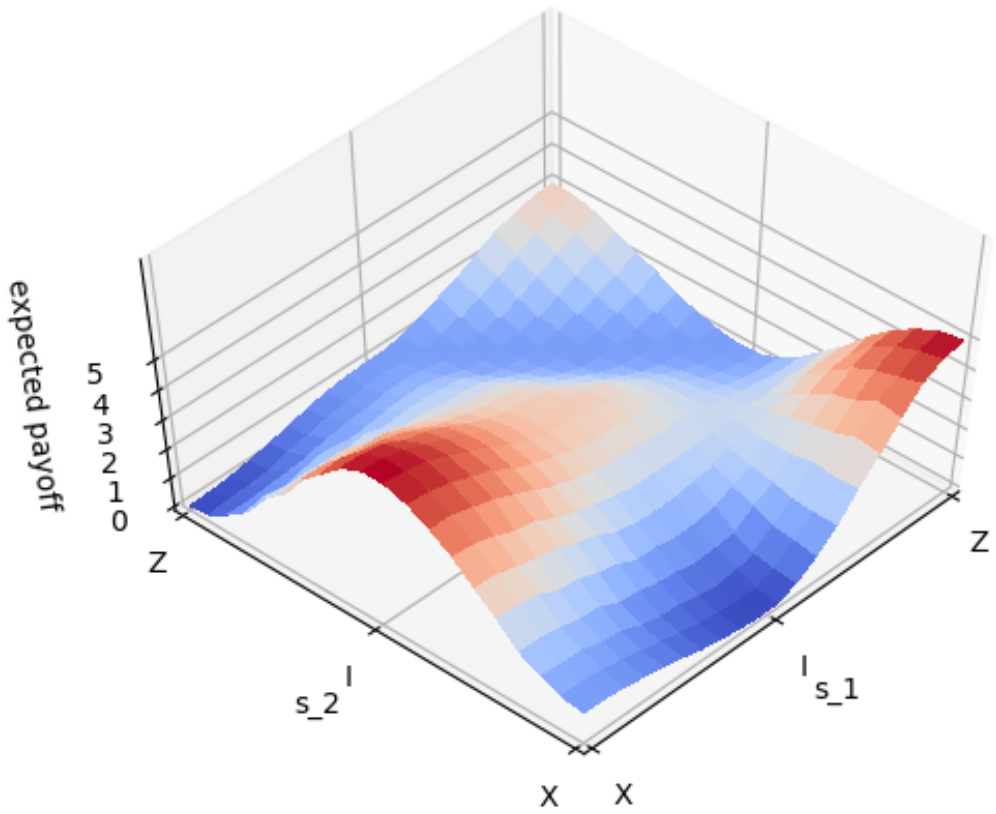

Figure 5.1. A graph of the payoff values for the first player which result from classically simulated games (on IBM Q simulator) where players employ strategies $f(s_1)$ and $f(s_2)$, respectively (see Eq. 5.2), and $s_1, s_2 \in [0, 1]$.

The graph in Figure 5.1 is consistent with the theoretical results. The part of the graph for strategies $X \to I$ is actually the graph for the mixed classical game and reproduces results from [26]. The chosen quantum extension shown in the remaining part of the graph contains interesting Z strategy acting as a switch of strategies between players. For example joint strategy (X, Z) gives the same payoff as joint strategy (I, X).

Table 5.2 below shows some statistical information about the results. The variance of the payoff values is always 0 for the simulated games due to the chosen set of strategies, each of which should yield exactly one outcome with probability 1 (thus having minimal entropy).

5.2. Quantum device

The next step was to execute the EWL Circuit on a real quantum device. This was done for three strategies: X, I, and Z. The Figure 5.2 below shows the payoff values for nine strategy profiles obtained by executing the EWL Circuit on the 16-qubit IBM Q Melbourne quantum device and applying the same expected payoff function (see Eq. 5.1). The measurement distribution π is known, as the experiment is prepared and executed on an IBM Q quantum device 1024 times (which is the default value on IBM Q).

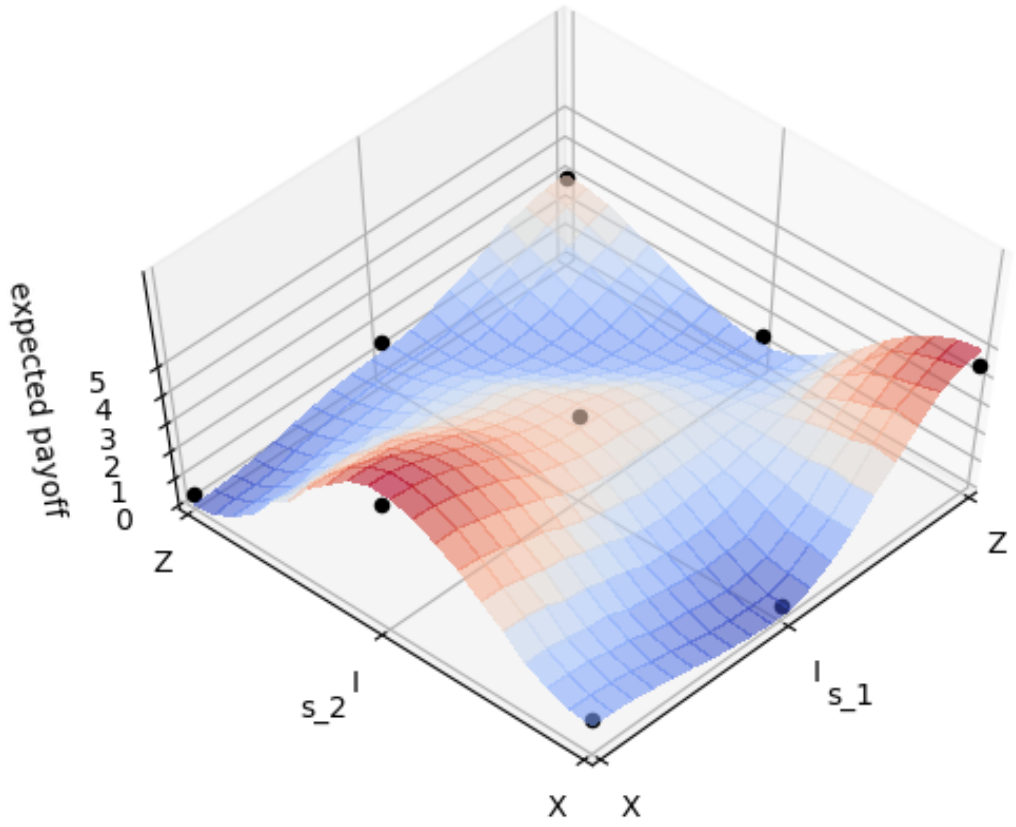

Figure 5.2. A graph of the payoff values for the first player from games executed on IBM Q Melbourne for strategies: X, I, and Z (the black dots). A semi-transparent surface of payoffs from games simulated classically (see Figure 5.1) is given for reference.

As we can see, the payoff graph is somewhat flattened when compared to the simulated results. This is due to the fact that quantum decoherence changes pure quantum states into mixed ones and thus produces measurement distributions with higher entropy. Table 5.2 below gives more statistical information about the results.

5.3. Quantum error correction

Another question of this thesis was, whether the results presented in the previous section can be improved. Two methods of quantum error correction were tested: the repetition code (for repetition number $d = 3$) and measurement error mitigation. These methods produced corrected measurement (outcome) distributions, to which the payoff function was also applied and the result is shown in the Table 5.2 below.

strategy profile	method	$E(p_A)$	$E(p_B)$	$V(p_A)$	$V(p_B)$
(X, X)	**simulation**	**1.0**	**1.0**	**0.0**	**0.0**
	quantum device	1.283	1.264	1.226	1.179
	repetition code	1.893	1.761	2.639	2.461
	meas. err. mitigation	1.0	1.0	0.0	0.0
(X, I)	**simulation**	**5.0**	**0.0**	**0.0**	**0.0**
	quantum device	4.494	0.422	1.725	1.383
	repetition code	3.303	1.35	4.061	3.382
	meas. err. mitigation	4.901	0.025	0.385	0.024
(X, Z)	**simulation**	**0.0**	**5.0**	**0.0**	**0.0**
	quantum device	0.444	4.502	1.272	1.49
	repetition code	2.093	2.708	3.627	3.75
	meas. err. mitigation	0.247	4.589	0.759	1.473
(I, X)	**simulation**	**0.0**	**5.0**	**0.0**	**0.0**
	quantum device	0.491	4.505	1.443	1.459
	repetition code	1.493	3.344	3.68	3.981
	meas. err. mitigation	0.298	4.6	0.994	1.432
(I, I)	**simulation**	**3.0**	**3.0**	**0.0**	**0.0**
	quantum device	2.879	2.947	0.827	0.771
	repetition code	2.628	2.789	1.648	1.581
	meas. err. mitigation	2.884	2.92	0.877	0.848
(I, Z)	**simulation**	**1.0**	**1.0**	**0.0**	**0.0**
	quantum device	1.224	1.346	1.158	1.455
	repetition code	2.174	2.223	3.771	3.8
	meas. err. mitigation	1.0	1.0	0.0	0.0
(Z, X)	**simulation**	**5.0**	**0.0**	**0.0**	**0.0**
	quantum device	4.504	0.437	1.539	1.297
	repetition code	2.39	2.004	3.603	3.369
	meas. err. mitigation	4.891	0.027	0.425	0.027

strategy profile	method	$E(p_A)$	$E(p_B)$	$V(p_A)$	$V(p_B)$
(Z, I)	**simulation**	**1.0**	**1.0**	**0.0**	**0.0**
	quantum device	1.216	1.382	1.118	1.517
	repetition code	2.229	2.292	3.983	4.013
	meas. err. mitigation	1.0	1.0	0.0	0.0
(Z, Z)	**simulation**	**3.0**	**3.0**	**0.0**	**0.0**
	quantum device	2.867	2.945	0.725	0.661
	repetition code	2.349	2.373	3.407	3.414
	meas. err. mitigation	2.868	2.916	0.743	0.705

Table 5.2. A table gathering expected payoffs and payoff variances for each player, strategy profile, and method. All values are rounded to the third decimal digit.

Table 5.2 shows that the repetition code not only did not improve the results but also made them significantly worse. This is probably due to the fact that the repetition code circuit involves a large number of quantum gates, especially CNOT gates, each of which introduces a substantial error. This error is to large for the repetition code to work properly. However, the second method of quantum error correction, measurement error mitigation, brought some major improvement. In some instances, where the simulated expected payoff was $(1, 1)$, the expected payoffs and variances for the corrected results were equal to the simulated values up to the third decimal digit.

In order to measure overall quality of the experimental results we can treat the tuple of expected payoffs for each method as a vector from a 9-dimensional vector space and find their Euclidean distance from the theoretical expected payoffs. Thus for each player i and method m we can define a value $D_{m,i}$:

$$D_{m,i} := \sqrt{\sum_{s \in \{X,I,Z\}^2} \left[(E(p_i))(s) - (E(p_{m,i}))(s)\right]^2}, \tag{5.3}$$

where $E(p_i)$ is the theoretical expected payoff function for player i and $E(p_{m,i})$ is the experimental expected payoff function for method m and player i. Table 5.3 below presents those values for each experimental method and player.

method	$D_{m,A}$	$D_{m,B}$
quantum device	1.0723	1.0965
repetition code	4.5333	4.2443
meas. err. mitigation	0.4497	0.5866

Table 5.3. The $D_{m,i}$ distances from theoretical expected payoffs for each method and player.

Table 5.3 shows that the repetition code method increased the distance from the theoretical results by a factor of above four and measurement error mitigation decreased the same distance by a factor of two.

5.4. Summary

We have presented the evaluation results of the EWL scheme realization. It showed that the EWL Circuit is valid and that results obtained on a real quantum device can be improved with some quantum error correction methods, in this case measurement error mitigation. We have also shown that the repetition code quantum error correction method was unsuccessful in this matter.

6. Summary and Conclusion

This chapter concludes this thesis with the summary of the achieved goals and some remarks concerning possible future works in this area.

6.1. Achieved Goals

This thesis shows that the realization of the EWL scheme on IBM Q quantum devices is possible by achieving the following goals stated in Chapter 1:

Quantum game realization on IBM Q

The EWL scheme has been formulated using the elementary quantum gates required by IBM Q and a quantum circuit has been constructed, which can be executed on IBM Q quantum devices. This is the first published realization of the EWL scheme on IBM Q and thus makes the quantum Prisoner's Dilemma available for playing by anyone with access to the Internet. It is also a full realization meaning that all possible quantum strategies can be employed.

Study on the influence of the quantum errors

The influence of the quantum errors on the game results has been studied and some statistical information is available in Tables 5.2 and 5.3.

Quantum error correction

Two methods of quantum error correction: the repetition code and measurement error mitigation have been studied and used to correct the outcomes of quantum plays on an IBM Q device. Their efficiency is shown in Tables 5.2 and 5.3. Measurement error mitigation turned out to have great results and the repetition code, at least in the form implemented in this thesis, failed to correct errors.

6.2. Future Works

For better insight on quantum errors state tomography should be performed to determine the quantum state produced by the EWL Circuit and to obtain the fidelity between experimental and theoretical

quantum states. One could also use the obtained quantum state and a defined measurement operator to calculate the payoff value (see Eq. 2.8) instead of using the measurement result distribution.

Another possible direction of future research could be to study more quantum error correction methods and their utility in the domain of quantum games. This topic was only touched in this thesis by the means of trial and error and a more thorough study is needed. In the future IBM Q devices may allow quantum gates after measurements and this would open the way to utilization of another kind of quantum error correction methods which require this condition.

List of Figures

List of Tables

Bibliography

[1] *IBM Makes Quantum Computing Available on IBM Cloud to Accelerate Innovation*. URL: *https://www-03.ibm.com/press/us/en/pressrelease/49661.wss* (visited on 2019-07-26).

[2] *The IBM Q Experience Web application*. URL: *https://quantum-computing.ibm.com/* (visited on 2019-07-26).

[3] *The Qiskit webpage*. URL: *https://qiskit.org/* (visited on 2019-05-28).

[4] D. A. Meyer. "Quantum strategies". In: *Phys. Rev. Lett.* 82 (1999), pp. 1052–1055.

[5] O. G. Zabaleta, J. P. Barrangú, and C. M. Arizmendi. "Quantum game application to spectrum scarcity problems". In: *Physica A: Statistical Mechanics and its Applications* 466 (2017), pp. 455–461.

[6] P. Frąckiewicz. "Application of the Eisert-Wilkens-Lewenstein quantum game scheme to decision problems with imperfect recall". In: *Journal of Physics A: Mathematical and Theoretical* 44, 325304 (2011).

[7] J. Eisert, M. Wilkens, and M. Lewenstein. "Quantum games and quantum strategies". In: *Phys. Rev. Lett.* 83 (1999), pp. 3077–3080.

[8] F. S. Khan et al. "Quantum games: a review of the history, current state, and interpretation". In: *Quantum Information Processing* 17, 309 (2018).

[9] K. R. Apt. "A Primer on Strategic Games". In: *Lectures in Game Theory for Computer Scientists*. Ed. by K. R. Apt and E. Grädel. Cambridge University Press, 2011. Chap. 1.

[10] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cornell University, 2010.

[11] E. W. Piotrowski and J. Sładkowski. "An invitation to quantum game theory". In: *International Journal of Theoretical Physics* 42 (2003), pp. 1089–1099.

[12] P. Zhang et al. "Optical realization of quantum gambling machine". In: *Europhysics Letters Association* 82, 30002 (2008).

[13] W. Balthazar et al. "Experimental realization of the quantum duel game using linear optical circuits". In: *Journal of Physics B: Atomic, Molecular and Optical Physics* 47, 47103544 (2015).

[14] J. Du et al. “Experimental realization of quantum games on a quantum computer”. In: *Phys. Rev. Lett.* 88, 137902 (2002).

[15] R. Prevedel et al. “Experimental realization of a quantum game on a one-way quantum computer”. In: *New Journal of Physics* 9, 205 (2007).

[16] A. R. C. Pinheiro et al. “Vector vortex implementation of a quantum game”. In: *Journal of the Optical Society of America B* 30, 3210 (2013).

[17] *The Quantum Coin Game in the Qiskit’s Github repository*. URL: *https://github.com/Qiskit/qiskit-tutorials/blob/master/community/games/Quantum-Coin-Game.ipynb* (visited on 2019-05-30).

[18] *The IBM Q webpage*. URL: *https://www.research.ibm.com/ibm-q/* (visited on 2019-05-22).

[19] A. Barenco et al. “Elementary gates for quantum computation”. In: *Phys. Rev. A* 52, 3457 (1995).

[20] *The quantum games directory in the Qiskit’s Github repository*. URL: *https://github.com/Qiskit/qiskit-tutorials/tree/master/community/games* (visited on 2019-05-30).

[21] *The Qiskit Terra documentation: Elementary Operations*. URL: *https://qiskit.org/documentation/terra/summary_of_quantum_operations.html* (visited on 2019-05-28).

[22] R. Myerson. *Game Theory. Analysis of Conflict*. Harvard University Press, 1997.

[23] K. Leyton-Brown and Y. Shoham. *Essentials of Game Theory. A Concise, Multidisciplinary Introduction*. Morgan & Claypool Publishers, 2008.

[24] A. Rapoport, A. Chammah, and C. Orwant. *Prisoner’s Dilemma: A Study in Conflict and Cooperation*. University of Michigan Press, 1965.

[25] S. Benjamin and P. Hayden. “Comment on “Quantum Games and Quantum Strategies””. In: *Phys. Rev. Lett.* 87, 069801 (2001).

[26] M. Szopa. “Dlaczego w dylemat więźnia warto grać kwantowo?” Polish. In: *Studia Ekonomiczne* 178 (2014), pp. 174–189.

[27] J. Koch et al. “Charge-insensitive qubit design derived from the Cooper pair box”. In: *Phys. Rev. A* 76, 042319 (2007).

[28] *IBM Q device information in Qiskit repository*. URL: *https://github.com/Qiskit/ibmq-device-information/* (visited on 2019-07-30).

[29] S. Tannu and M. Qureshi. “Not All Qubits Are Created Equal: A Case for Variability-Aware Policies for NISQ-Era Quantum Computers”. In: 2019.

[30] *A comparison of metrics concerning qubit quality*. URL: *https://quantumcomputingreport.com/scorecards/qubit-quality/* (visited on 2019-08-22).

[31] E. Knill et al. “Randomized benchmarking of quantum gates”. In: *Phys. Rev. A* 77, 012307 (2008).

[32] M. Bossert. *Channel Coding for Telecommunications*. Wiley, 1999.

[33] J. Wootton and D. Loss. “Repetition code of 15 qubits”. In: *Phys. Rev. A* 97, 052313 (2018).

[34] P. Murali et al. “Noise-Adaptive Compiler Mappings for Noisy Intermediate-Scale Quantum Computers”. In: (2019).

[35] V. Shende, S. Bullock, and I. Markov. “Synthesis of quantum-logic circuits”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25.6 (2006), pp. 1000–1010.

[36] C. Li, R. Roberts, and X. Yin. “Decomposition of unitary matrices and quantum gates”. In: *International Journal of Quantum Information* 11, 1350015 (2013).

[37] *Matlab Program pud.m written by Rebecca Roberts*. URL: *https://cklixx.people.wm.edu/mathlib.html* (visited on 2019-09-12).