A study of one-turn quantum refereed games

by

Soumik Ghosh

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science (Quantum Information)

Waterloo, Ontario, Canada, 2020

Author's declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis studies one-turn quantum refereed games, which are abstract zero-sum games with two competing computationally unbounded quantum provers and a computationally bounded quantum referee. The provers send quantum states to the referee, who plugs the two states into his quantum circuit, measures the output of the circuit in the standard basis, and declares one of the two players as the winner depending on the outcome of the measurement. The complexity class QRG(1) comprises of those promise problems for which there exists a one-turn quantum refereed game such that one of the players wins with high probability for the yes-instances, and the other player wins with high probability for the no-instances, irrespective of the opponent's strategy. QRG(1) is a generalization of QMA (or co-QMA), and can informally be viewed as QMA with a no-prover (or co-QMA with a yes-prover).

We have given a full characterization of QRG(1), starting with appropriate definitions and known results, and building on to two new results about this class. Previously, the best known upper bound on QRG(1) was PSPACE. We have proved that if one of the provers is completely classical, sending a classical probability distribution instead of a quantum state, the new class, which we name CQRG(1), is contained in $\exists \cdot \mathrm{PP}$ (nondeterministic polynomial-time operator applied to the class PP). We have also defined another restricted version of QRG(1) where both provers send quantum states, but the referee measures one of the quantum states first, and plugs the classical outcome into the measurement, along with the other prover's quantum state, into a quantum circuit, before measuring the output of the quantum circuit in the standard basis. The new class, which we name MQRG(1), is contained in $\mathrm{P} \cdot \mathrm{PP}$ (the probabilistic polynomial time operator applied to PP). $\exists \cdot \mathrm{PP}$ is contained in $\mathrm{P} \cdot \mathrm{PP}$, which is, in turn, contained in PSPACE. Hence, our results give better containments than PSPACE for restricted versions of QRG(1).

Acknowledgements

I was very fortunate to have been blessed with an excellent set of mentors and friends. To them, I convey my heartfelt gratitude. I came to Waterloo scared and unsure, having just completed my undergraduate and travelling abroad for the first time in my life. I come out a little more confident in my abilities as a researcher, having spent two very memorable years of my life here. Although this section is shorter in length to the other ones and does not have mathematical equations of interest to scientists, it captures one of the most important components of my journey: the human component.

There are many who helped me make the last two years of my life joyful. Of them, the most important influence was, undoubtedly, my supervisor John Watrous. John was a father-figure who was always there, through my highs and lows. I knew very little, if anything, about quantum computing when I came to Waterloo. John taught me like a parent teaches someone how to ride a bike. Whenever I tumbled, I knew John was there to immediately lend me a hand. He had infinite patience for my silly questions about research, for my philosophical musings about life, and for everything in between. He is an excellent human being, a near perfect supervisor, an excellent lecturer, and an excellent researcher. His truly fascinating mathematical insights, for even the most abstract equations, are matched by his equally amazing sense of humour. Watching John teach is like watching a Wes Anderson movie; with the auteur in complete control of his craft, the experience is wholesome and you come out feeling content. John will always be so much more than an academic mentor for me, and I was fortunate enough to have ever met him. To shamelessly borrow a quote from one of his students on ratemyprofessors, "He is quite possibly the ruler of the universe...in disguise."

I am also grateful to many other mentors whom I met along my journey. Of them, I am very thankful to David Gosset. David was always a knock-on-the-door away from anything I wanted to ask. He was kind enough to let me attend his group meetings, friendly enough to always have time for a chat in the atrium, and patient enough to always entertain my research-ideas, which were often half-baked and not very coherent. Fridays will never be the same without his group meetings. I had many inspiring interactions with David. It is not hard to gauge from them that he is a great researcher, an amazing mentor, and a true "friend, guide, and philosopher" to his students.

There are two others I have to specifically thank for always being there for me. One of them is Matthew Coudron, with whom I had countless hours of discussions, often over gourmet buffet dinner at St Jerome's. The other is Jamie Sikora, who is one of the friendliest and most energetic researchers I have ever met, and someone I could talk to for hours about anything from QMA(2) to animated Batman movies. They are both wonderful people and great researchers. My deep thanks to both of them for helping me grow as a researcher and also for treating me as a friend. My experience would have been so much duller without them.

Besides research, I am grateful to Rajibul Islam to let me be a part of "bigyan". Popularizing science in Bengali is an extremely noble initiative. I hope his endeavour continues

to inspire many more students, like me.

Just like mentors, I was blessed to have an equally fantastic set of friends at IQC. My very special thanks to Aditya and Karen. I cannot imagine how life must have been if I had not met you guys. From research to everything else on earth, there's probably nothing we did not talk about. I will always cherish the very colourful memories: from sweating the assignments out in John's course, debating philosophy and world politics far past bedtime, to doing karaoke, meeting Scott, roaming in Vancouver, and so much more. Thanks for always being there and hope to have many more memories!

There are many others at IQC who made my life joyful. In no particular order, thanks to Lane, Ian, Supratik, and Vinodh. They are friends whom I will always treasure as a source of support. I will miss Lane's rollicking sense of humour and Ian's perspicacious wisdom. I will miss the discussions, often deep and personal, with both of them, about life and research! I will miss chugging beers with Vinodh at monthly gatherings, and going down in the elevator and talking to Supratik about anything at all, anytime I felt I needed someone to talk to. I also miss sneaking up on free food at IQC conferences with a subset of them, during different times of the year. I hope Lane, Ian, and Supratik turn out to be amazing researchers, and Vinodh finds his perfect job in the industry.

There are many in John's group who deserve special mention. Thanks to Colin for countless hours of discussions, often futile but always fun, about life and research, and for being a great friend throughout! Thanks to Abel for being a source of infinite wisdom about life, research, graduate school, and beyond, for being someone I could talk to for hours about shared interests like politics, and for being a wonderful TA. Thanks to Sanketh for entertaining my extremely silly questions whenever I asked them, and for good memories from the trip to Toronto!

I am just as thankful to others I met at parties, monthly gatherings, pizza nights, and movie nights. Thanks to Dan for always driving me around Waterloo and for fun memories at Sherbrooke! Thanks to Jimmy for enlivening my stay at Sherbrooke! Thanks to Nicky for agonizing over grad school and grad school rejections together and for very interesting discussions during cookie time! Thanks to Shima and Meenu for being wonderful officemates! Thanks to Rahul and Vadiraj, for very cordially inviting me to their home countless times, for everything from poetry sessions and movie nights to carrom matches and Diwali parties! I will miss the walk home, in the middle of the night, from Albert Street to Cedarvale.

Just like my friends at IQC, I will miss my friends in Computer Science. Thanks to Sushant for being a great friend and a great roommate in Toronto! Thanks to Zeynep for our extremely refreshing discussions over lunch or coffee!

I was blessed with amazing roommates, who made my stay much more pleasurable than it would have been, especially during the pandemic. Thanks to Rishiraj for helping me out when I knew nothing about Canada or Waterloo, and for being witness to many happy memories in the house for one year. Thanks to Saptarshi, whom I have known for seven years now, right from high school. I have not, and probably will not meet, someone with more expertise in movies who's also a brilliant researcher and a great friend. Thanks

to Manas for being like an elder brother, right from the very day I met him at IQC, and for his infectious love of all things scientific. I hope I imbibed some of it from him. Thanks to Sourya, for being one of the most versatile and most talented persons I have ever met, with wide-ranging interests in research, in film-making, in drama, in recitation, and in singing. I will miss our colourful banter, our debates, and so much more. All of you made me feel like home thousands of miles away from home. Our house at Cedarvale was the witness to many wonderful memories, from watching movies together, to going to groceries together, cooking together (which I mostly made a mess of!), and having countless discussions about everything under the sun. I will miss you all wherever I am.

I am also grateful to my friends from undergraduate. I am grateful to Arnab, for being a friend who always lent the listening ear, for introducing me to quantum computing in the first place, for having so many memories during undergraduate including doing research together, and also for showing me around in Vancouver. I am grateful to Shomik, for being understanding and for putting up with me for all these years. I am grateful to Dripta, Purbesh, and Pijush, for being amazing friends, through thick and thin. Thanks to Debanjan for fun memories whenever I returned home and for endless discussions over social media. Thanks to Henry for staying in touch for four years, even after being geographically separated across two different continents, for extremely insightful discussions that enriched me as an academic and as a person, and for great memories from the visit to London! Even though it has been two years since my undergraduate, I will always be indebted to all of these people for their invaluable friendship.

Finally, I must thank my parents, without whom none of this would have been possible. It was hard for them to have me leave home for the first time, but they never showed it and never stopped supporting me, through happy and tumultuous times. They made great sacrifices so that I can be where I am, and I am forever grateful to them for their unconditional love. This thesis is as much theirs as it is mine.

Dedication

This is dedicated to my parents.

Table of Contents

List of Figures

Chapter 1

Introduction

Games have fascinating links to theoretical computer science. Many popular games can be formulated as complete problems for complexity classes. For example, Minesweeper is NP-complete [Kay00], Rush Hour is PSPACE-complete [FB02], and Chess is EXP-complete [FL81]. Games are intricately linked with the alternating Turing machine model of computation [CKS81] from computability theory. Games have been used to define new complexity classes: Arthur Merlin games [Bab85, BM88] define the complexity class AM. Interactive proof systems [GMR85, GMR89], which generalize prover-verifier interactions of the type seen in AM, can also be naturally formulated in terms of games [Con87, FKS95].

Quantum refereed games are played between two competing, computationally unbounded quantum provers who exchange quantum messages with a computationally bounded quantum referee. We name the two players *Alice* and *Bob*. The complexity class QRG is defined as the set of promise problems $A = (A_{\text{yes}}, A_{\text{no}})$ for which there exists a quantum refereed game such that Alice wins with high probability for $x \in A_{\text{yes}}$ and Bob wins with high probability for $x \in A_{\text{no}}$, regardless of the other player's strategy. The complexity class RG is defined similarly, with classical refereed games. The players' strategies can be described by probability distributions in the classical case and density matrices in the quantum case. RG was investigated in the 1990s by Koller and Megiddo [KM92], Feigenbaum, Koller, and Shor [FKS95], Condon, Feigenbaum, Lund, and Shor [CFLS95, CFLS97], and Feige and Kilian [FK97]. In particular, Feige and Kilian showed that RG is equal to EXP. QRG was later considered in [GW05], [Gut05], [GW07], and [JW09]. It was shown in [GW07] that QRG, just like its classical analogue, is equal to EXP.

This thesis is concerned with quantum refereed games with only *one turn* of interaction between the players and the referee. This means the players and the referee first receive a common input string, the two players then send two polynomial length quantum or classical proofs to the referee, and the referee finally processes the two proofs to decide which player has won. QRG(1) and RG(1) are defined similar to QRG and RG, with *one-turn* quantum and classical refereed games respectively. Note that this naming clashes with [FK97], which defines RG(1) as *one-round* (i.e., two-turn) refereed games, which is RG(2) in terms of our naming conventions. An informal pictorial description of

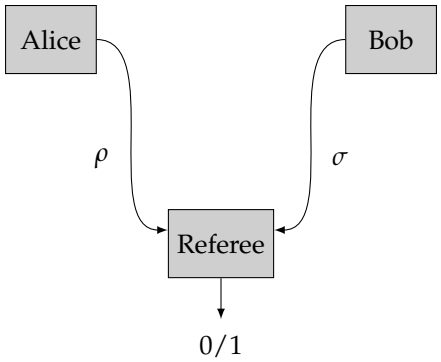

Figure 1.1: A pictorial description of QRG(1). Alice and Bob send density matrices ρ and σ respectively to the referee. The referee processes the two messages and decides who wins the game, by outputting the classical bit 1 if Alice wins and the classical bit 0 if Bob wins.

the setting in QRG(1) follows in Figure 1.1. Although it is not stated explicitly, it follows from the works of Althöfer [Alt94] and Lipton and Young [LY94] that the complexity class $\mathrm{RG}(1)$ is equal to S_2^p, which refers to the second level of the *symmetric polynomial-time hierarchy* put forward by Canetti [Can96] and Russell and Sundaram [RS98]. S_2^p is defined in terms of games with two players where the players send polynomial length strings to a deterministic referee. The equivalence means that S_2^p does not change if the players are allowed to choose probability distributions instead of strings and the referee is allowed to use randomness. As stated earlier, a way to see this equivalence is the Althöfer-Lipton-Young method which involves arguing for the existence of a near-optimal strategy for non-interactive randomized two player zero sum games which is uniform over a polynomial sized set of polynomial length strings. It is also known that $\mathrm{RG}(1)$ is closed under Cook reductions [RS98] and satisfies $\mathrm{RG}(1) \subseteq \mathrm{ZPP}^{\mathrm{NP}}$ [Cai07].

On the other hand, for the complexity class $\mathrm{QRG}(1)$, the only non-trivial containment known is that $\mathrm{QRG}(1) \subseteq \mathrm{PSPACE}$ [JW09]. We know slightly more about $\mathrm{QRG}(2)$, the two-turn analogue of $\mathrm{QRG}(1)$. From the result of [GW13], $\mathrm{QRG}(2)$ equals PSPACE. $\mathrm{QRG}(1)$ can be thought of as a generalization of QMA, with a no prover in addition to a yes prover. It is easy to see that $\mathrm{QRG}(1)$ contains $\mathrm{QMA} \cup \mathrm{coQMA}$.

1.1. Summary of the results

In this thesis, we study the power of two restricted variants of $\mathrm{QRG}(1)$ and prove a containment better than PSPACE for these variants. In the first variant, Alice is restricted to choosing a classical probability distribution while Bob is free to play a quantum state. We call the corresponding class $\mathrm{CQRG}(1)$ and prove that it is contained in $\exists \cdot \mathrm{PP}$ (the non-deterministic polynomial time operator applied to the complexity class PP). In the second variant, Alice is free to send a quantum state but the referee measures Alice's state

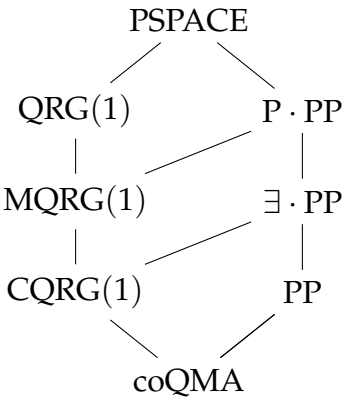

Figure 1.2: Hasse diagram showing the relationships between CQRG(1), MQRG(1), and other complexity classes.

before incorporating the classical outcome of the measurement into a subsequent measurement of Bob's state. We call the corresponding class MQRG(1) and prove that it is contained in P · PP (the unbounded error probabilistic polynomial time operator applied to the complexity class PP). Our results are pictorially represented in Figure 1.2, with a Hasse diagram. A detailed definition of ∃ · PP and PP · PP will follow in later chapters. Our proofs generalize the Althöfer-Lipton-Young method for quantum settings, by making use of tail bounds for random matrices [Tro12].

Observing that ∃ · PP and P · PP are contained in the counting hierarchy, we conjecture that QRG(1) is also contained in the counting hierarchy.

1.2. Overview

In this section, we give a bird's eye view of the rest of the chapters.

- In Chapter 2, we introduce the basic notions of complexity theory, including concepts from counting complexity.
- In Chapter 3, we introduce the basics of quantum information and relate them with complexity theory.
- In Chapter 4, we give rigorous definitions of all the complexity classes under consideration.
- In Chapter 5, we prove a folklore result: $\mathrm{QRG}(1) = \mathrm{P}^{\mathrm{QRG}(1)}$. We also prove a containment result: $\mathrm{CQRG}(1) \subseteq \exists \cdot \mathrm{PP}$.
- In Chapter 6, we prove another containment result: $\mathrm{MQRG}(1) \subseteq \mathrm{P} \cdot \mathrm{PP}$.
- In Chapter 7, we conclude with future directions.

The proofs of the two containments $\mathrm{CQRG}(1) \subseteq \exists \cdot \mathrm{PP}$ and $\mathrm{MQRG}(1) \subseteq \mathrm{P} \cdot \mathrm{PP}$ appear in the following paper:

- Soumik Ghosh and John Watrous. Complexity limitations on one-turn quantum refereed games, 2020.

Chapter 2

Complexity theory

In this chapter, we give an overview of parts of complexity theory that are relevant for the rest of the theses. It is not meant to be exhaustive; it is only aimed at giving a unifying picture of the rest of the thesis. We refer the interested reader to books by Arora and Barak [AB09] or Goldreich [Gol08] for a detailed study in the concepts dealt with here.

The first section deals with notation, terminology, and the basics of complexity theory. The second section deals with a few concepts from counting complexity that we generalize. We will find these generalized concepts very useful while proving the main theorems of the thesis.

2.1. Basics of complexity theory

As a convention, we use Greek capital letters (like Σ, Δ, etc) to refer to finite, non-empty sets of symbols called *alphabets*. The set of natural numbers, including 0, is denoted $\mathbb{N}$. When we are referring to the set $\{1, 2, \ldots, n\}$, for some $n \in \mathbb{N}$, we will often use the shorthand $[n]$. Moreover, we will use the binary alphabet $\Sigma = \{0, 1\}$ throughout the thesis. We will also use Σ_1^n to denote the set of all strings over the binary alphabet Σ that have length n and contain exactly one occurrence of the symbol 1. It is therefore the case that $|\Sigma_1^n| = n$.

A *polynomially bounded* function $p : \mathbb{N} \to \mathbb{N}$ is a function for which there exists a deterministic Turing machine which runs in polynomial time and outputs $0^{p(n)}$ on input 0^n for all $n \in \mathbb{N}$. Unless it is explicitly indicated otherwise, the input of a given polynomially bounded function p is assumed to be the natural number $|x|$, for whatever input string $x \in \Sigma^*$ is being considered at that moment. With this in mind, we will write p in place of $p(|x|)$ when referring to the natural number that is the output for p. This clears clutter in notation. For example, we have used this convention in Definition 1 below and in many places in the thesis thereafter.

A *promise problem* is a pair $A = (A_{\text{yes}}, A_{\text{no}})$ of sets of strings $A_{\text{yes}}, A_{\text{no}} \subseteq \Sigma^*$ with $A_{\text{yes}} \cap A_{\text{no}} = \varnothing$. Strings in A_{yes} represent yes-instances of a decision problem, strings in A_{no} represent no-instances, and all other strings represent "don't care" inputs for which

we do not care about whether it is a yes-instance or a no-instance. A *language* is a promise problem $A = (A_{\text{yes}}, A_{\text{no}})$ where we require $A_{\text{yes}} \cup A_{\text{no}} = \Sigma^*$. In other words, languages are promise problems where do not have the "don't care" inputs. A language A is often identified with A_{yes} by dropping the subscript *yes* with the implicit assumption that $A_{\text{no}} = \Sigma^* \setminus A_{\text{yes}}$. A *complexity class* of languages is a set of languages. A *characteristic function* of a language A is a function $\chi_A : \Sigma^* \to \Sigma$ that is defined as

$$\chi_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A, \end{cases} \tag{2.1}$$

We will often refer to two reductions between inputs of two languages. In a *Karp reduction* from a language A to a language B, there exists a polynomial-time computable function g for transforming inputs $x \in \Sigma^*$ of language A to inputs $y \in \Sigma^*$ of language B such that $x \in A$ if and only if $y \in B$. In a *Cook reduction* from a language A to a language B, there exists a polynomial-time oracle Turing machine M such that if N is any Turing machine that decides B, then M^N decides A. In a polynomial-time *truth-table reduction* from a language A to a language B, there exists a polynomial-time computable function g for transforming inputs $x \in \Sigma^*$ of language A into a fixed number of inputs $y_1, \ldots, y_k \in \Sigma^*$ of language B, and a polynomial-time computable function $f : \{0,1\}^k \to \{0,1\}$ such that $x \in A$ if and only if

$$f\big(\chi_B(y_1), \ldots, \chi_B(y_n)\big) = 1 \tag{2.2}$$

where χ_B is the characteristic function for language B.

Another operation on languages, called *join*, will also be used in the next sections. The *join* of languages A and B is defined as $\{x0 : x \in A\} \cup \{x1 : x \in B\}$.

We fix a pairing function that efficiently encodes two strings $x, y \in \Sigma^*$ into a single binary string denoted $\langle x, y \rangle \in \Sigma^*$, and we assume that this function satisfies two simple properties:

1. The length of the pair $\langle x, y \rangle$ depends only on the lengths $|x|$ and $|y|$, and is polynomial in these lengths.
2. The computation of x and y from $\langle x, y \rangle$, as well as the computation of $\langle x, y \rangle$ from x and y, can be performed deterministically in polynomial time.

One such choice of function is as follows:

$$\langle a_1 a_2 \cdots a_n, b_1 b_2 \cdots b_m \rangle = 0a_1 0a_2 \cdots 0a_n 1b_1 b_2 \cdots b_m \tag{2.3}$$

for $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m \in \Sigma$. This pairing function may be extended recursively as follows:

$$\langle x_1, x_2, x_3, \ldots, x_k \rangle = \langle \langle x_1, x_2 \rangle, x_3, \ldots, x_k \rangle \tag{2.4}$$

for strings $x_1, \ldots, x_k \in \Sigma^*$, where $k \geq 3$. From now on, whenever we refer to the computation of any function that takes multiple strings as arguments, we make the implicit

assumption that these input strings have been encoded into a single string using this tuple function. For instance, when f is a function that takes three binary strings x, y, and z as arguments, we write $f(x, y, z)$ rather than $f(\langle x, y, z\rangle)$.

Next, we define the nondeterministic and probabilistic polynomial-time operators. These operators may be applied to arbitrary complexity classes.

Definition 1. For a given complexity class of languages $\mathcal{C}$, the complexity classes $\exists \cdot \mathcal{C}$ and $\mathrm{P} \cdot \mathcal{C}$ are defined as follows.

1. The complexity class $\exists \cdot \mathcal{C}$ contains all promise problems $A = (A_{\text{yes}}, A_{\text{no}})$ for which there exists a language $B \in \mathcal{C}$ and a polynomially bounded function p such that the two implications below are true:

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in B \right\} \neq \varnothing, \\ x \in A_{\text{no}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in B \right\} = \varnothing. \end{aligned} \tag{2.5}$$

2. The complexity class $\mathrm{P} \cdot \mathcal{C}$ contains all promise problems $A = (A_{\text{yes}}, A_{\text{no}})$ for which there exists a language $B \in \mathcal{C}$ and a polynomially bounded function p such that the two implications below are true:

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left| \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in B \right\} \right| > \frac{1}{2} \cdot 2^p, \\ x \in A_{\text{no}} &\Rightarrow \left| \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in B \right\} \right| \leq \frac{1}{2} \cdot 2^p. \end{aligned} \tag{2.6}$$

2.2. Counting complexity

In this section, we will state some results which are minor generalizations of a few well known results from counting complexity. It is not meant to be an exhaustive introduction to counting complexity. Readers interested in the same should refer to Fortnow's survey paper [For97]. We will begin with a definition.

Definition 2. Let $\mathcal{C}$ be any complexity class of languages over the alphabet Σ. A function $f : \Sigma^* \to \mathbb{Z}$ is a $\mathrm{Gap} \cdot \mathcal{C}$ function if there exist languages $A, B \in \mathcal{C}$ and a polynomially bounded function p such that

$$f(x) = \left| \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in A \right\} \right| - \left| \left\{ y \in \Sigma^p \,:\, \langle x, y\rangle \in B \right\} \right| \tag{2.7}$$

for all $x \in \Sigma^*$.

Observe that gap functions are usually defined in terms of the difference between the number of accepting and rejecting paths of a non-deterministic Turing machine and our

definition of gap functions is slightly non-standard. However, note that the two definitions are equivalent when $\mathcal{C} = \mathrm{P}$ [1], and for that case, we write GapP instead of Gap $\cdot$ P in order to be consistent with the standard name for these functions. One way to see the equivalence is to note that GapP can also be defined as the difference between the accepting paths of two different non-deterministic Turing machines (instead of the difference between the accepting and rejecting paths of the same machine). For a better elucidation of this fact, we direct the reader to Proposition 3.5 of [FFK94] or Lemma 3.6 of Fortnow's survey [For97]. Later in the thesis, we will also consider Gap $\cdot$ PP, which will be defined according to Definition 2, with $\mathcal{C} = \mathrm{PP}$.

Just like GapP functions, Gap $\cdot\, \mathcal{C}$ functions have strong closure properties provided $\mathcal{C}$ itself also has some closure properties. For the closure properties that we need, it suffices for $\mathcal{C}$ to be nontrivial, which means that $\mathcal{C}$ contains at least one language that is not equal to $\varnothing$ or Σ^*) and is closed under the join operation as well as polynomial-time truth-table reductions; both these operations were defined in the previous section. It can be worked out that the fact $\mathcal{C}$ is closed under joins and polynomial-time truth table reductions implies that $\mathcal{C}$ is closed under well-known properties like unions, intersections, and complementation. These properties are possessed by both P and PP. The fact that PP is closed under polynomial-time truth table reductions was proved by Fortnow and Reingold [FR96] based on methods found in [BRS95]. We will soon state the closure properties of Gap $\cdot\, \mathcal{C}$. Before that, we state and prove a proposition relating the class P $\cdot\, \mathcal{C}$ to Gap $\cdot\, \mathcal{C}$ functions.

Proposition 3. *Let $\mathcal{C}$ be a complexity class of languages that is closed under join and complementation. A promise problem $A = (A_{yes}, A_{no})$ is contained in* P $\cdot\, \mathcal{C}$ *if and only if there exists a* Gap $\cdot\, \mathcal{C}$ *function f such that*

$$\begin{aligned} x \in A_{yes} &\Rightarrow f(x) > 0, \\ x \in A_{no} &\Rightarrow f(x) \leq 0. \end{aligned} \tag{2.8}$$

Proof. Let us assume that a promise problem A is contained in P $\cdot\, \mathcal{C}$. This means there exists a $B \in C$ such that Definition 1 holds. With the understanding that $\overline{B}$ is contained in $\mathcal{C}$, define $f(x)$ as

$$f(x) = \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B\right\}\right| - \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in \overline{B}\right\}\right| \tag{2.9}$$

for all $x \in \Sigma^*$. It may be observed that

$$\left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B\right\}\right| + \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in \overline{B}\right\}\right| = 2^p \tag{2.10}$$

[1] The P here refers to the class of promise problems solvable in polynomial-time by a deterministic Turing machine. This is different from the probabilistic polynomial time operator of Definition 1. In general, for the rest of this thesis, if the symbol P precedes a "$\cdot$", interpret it as a probabilistic polynomial time operator. Otherwise, interpret it as a complexity class of promise problems solvable in polynomial time by a deterministic Turing machine.

for all $x \in \Sigma^*$. Then, it immediately follows that

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow f(x) > 0, \\ x \in A_{\text{no}} &\Rightarrow f(x) \leq 0. \end{aligned} \tag{2.11}$$

For the other direction, assume that for a promise problem A, there exists a Gap $\cdot\, \mathcal{C}$ function f and a polynomially bounded function p such that

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow f(x) > 0, \\ x \in A_{\text{no}} &\Rightarrow f(x) \leq 0. \end{aligned} \tag{2.12}$$

Consider that $f(x)$ is given by

$$f(x) = \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B_1\right\}\right| - \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B_2\right\}\right| \tag{2.13}$$

for all $x \in \Sigma^*$ for $B_1, B_2 \in \mathcal{C}$. Consider the language C where C is the join of B_1 and $\overline{B_2}$. Since $\mathcal{C}$ is closed under complementation and joins, we have $C \in \mathcal{C}$. It can be observed that

$$\begin{aligned} &\left|\left\{y \in \Sigma^{p+1} \,:\, \langle x, y\rangle \in C\right\}\right| \\ &= \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B_1\right\}\right| + 2^p - \left|\left\{y \in \Sigma^p \,:\, \langle x, y\rangle \in B_2\right\}\right| = 2^p + f(x) \end{aligned} \tag{2.14}$$

for all $x \in \Sigma^*$. Taking C as our language and $p+1$ as our polynomially bounded function, it follows from equation (2.11) that

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left|\left\{y \in \Sigma^{p+1} \,:\, \langle x, y\rangle \in C\right\}\right| > \frac{1}{2} \cdot 2^{p+1}, \\ x \in A_{\text{no}} &\Rightarrow \left|\left\{y \in \Sigma^{p+1} \,:\, \langle x, y\rangle \in C\right\}\right| \leq \frac{1}{2} \cdot 2^{p+1}. \end{aligned} \tag{2.15}$$

□

Next, we state and prove a few closure properties for GapP functions. The proof of the first lemma utilizes the fact that $\mathcal{C}$ is closed under Karp reductions.

Lemma 4. *Let $\mathcal{C}$ be a nontrivial complexity class of languages that is closed under Karp reductions. Let $f \in$ Gap $\cdot\, \mathcal{C}$ and let p be a polynomially bounded function. The function*

$$g(x) = \sum_{y \in \Sigma^p} f(x, y) \tag{2.16}$$

is a Gap $\cdot\, \mathcal{C}$ *function.*

Proof. Note that $f \in$ Gap $\cdot\, \mathcal{C}$ means that we can find a polynomially bounded function q and languages $A_0, A_1 \in \mathcal{C}$ such that

$$f(x, y) = \left|\left\{z \in \Sigma^{q(|\langle x,y\rangle|)} \,:\, \langle x, y, z\rangle \in A_0\right\}\right| - \left|\left\{z \in \Sigma^{q(|\langle x,y\rangle|)} \,:\, \langle x, y, z\rangle \in A_1\right\}\right| \tag{2.17}$$

for all $x \in \Sigma^*$ and $y \in \Sigma^p$. Taking into account the assumptions on the pairing function described in the last section, we can argue that $|\langle x, y\rangle|$ depends only on $|x|$ and $|y|$. Therefore, there exists a function r such that $r(|x|) = p(|x|) + q(|\langle x, y\rangle|)$ for all $x \in \Sigma^*$ and $y \in \Sigma^p$. Since p and q are polynomially bounded functions, r is also a polynomially bounded function. Define

$$\begin{aligned} B_0 &= \left\{ \langle x, yz\rangle \,:\, y \in \Sigma^p,\ z \in \Sigma^{q(|\langle x,y\rangle|)},\ \langle x, y, z\rangle \in A_0 \right\}, \\ B_1 &= \left\{ (x, yz) \,:\, y \in \Sigma^p,\ z \in \Sigma^{q(|\langle x,y\rangle|)},\ \langle x, y, z\rangle \in A_1 \right\}. \end{aligned} \tag{2.18}$$

By the nontriviality and closure of $\mathcal{C}$ under Karp reductions, it is evident that $B_0, B_1 \in \mathcal{C}$. To illustrate on the choice of function for the Karp reduction, note that with our choice of the efficient pairing function, there is a polynomial time computable function that takes $\langle x, yz\rangle$ to $\langle x, y, z\rangle$. It may be verified that

$$g(x) = \left|\left\{w \in \Sigma^r \,:\, \langle x, w\rangle \in B_0\right\}\right| - \left|\left\{w \in \Sigma^r \,:\, \langle x, w\rangle \in B_1\right\}\right| \tag{2.19}$$

for all $x \in \Sigma^*$, and therefore $g \in \mathrm{Gap} \cdot \mathcal{C}$. □

The proof of the second lemma utilizes the full power of $\mathcal{C}$, requiring it to be closed under joins and polynomial-time truth table reductions.

Lemma 5. *Let $\mathcal{C}$ be a nontrivial complexity class of languages that is closed under joins and polynomial-time truth table reductions. Let $f \in \mathrm{Gap} \cdot \mathcal{C}$ and let p be a polynomially bounded function. The function*

$$g(x) = \prod_{y \in \Sigma_1^p} f(x, y) \tag{2.20}$$

is a $\mathrm{Gap} \cdot \mathcal{C}$ *function.*

Proof. Since $f \in \mathrm{Gap} \cdot \mathcal{C}$, there exists a polynomially bounded function q and languages $A_0, A_1 \in \mathcal{C}$ such that

$$f(x, y) = \left|\left\{z \in \Sigma^{q(|(x,y)|)} \,:\, \langle x, y, z\rangle \in A_0\right\}\right| - \left|\left\{z \in \Sigma^{q(|(x,y)|)} \,:\, \langle x, y, z\rangle \in A_1\right\}\right| \tag{2.21}$$

for all $x, y \in \Sigma^*$. Without loss of generality, we can assume that A_0 and A_1 are disjoint languages. If they are not, we may replace A_0 and A_1 with $A_0 \cap \overline{A_1}$ and $A_1 \cap \overline{A_0}$, respectively; this does not change the value of the right-hand side of the equation (2.21) as the strings that are common to A_0 and A_1 get subtracted anyway in equation (2.21), and hence do not contribute to the value of $f(x, y)$. The languages $A_0 \cap \overline{A_1}$ and $A_1 \cap \overline{A_0}$ must both be contained in $\mathcal{C}$ for $A_0, A_1 \in \mathcal{C}$ by the closure of $\mathcal{C}$ under joins and truth-table reductions (which implies that it is closed under intersection and complementation).

By the assumptions on our pairing function described in the last section, there exists a polynomially bounded function r such that $r(|x|) = q(|(x, y)|)$ for all $x \in \Sigma^*$ and $y \in \Sigma^p$. We will write $y_1, \ldots, y_p$ to denote the elements of Σ_1^p sorted in lexicographic order. Let us define two languages B_0 and B_1 as follows:

- B_0 is the language of all pairs $\langle x, z_1 \cdots z_p \rangle$, where $x \in \Sigma^*$ and $z_1, \ldots, z_p \in \Sigma^r$, for which there exists a string $w \in \Sigma^p$ having even parity such that

$$\langle x, y_1, z_1 \rangle \in A_{w_1}, \ldots, \langle x, y_p, z_p \rangle \in A_{w_p}. \tag{2.22}$$

- B_1 is the language of all pairs $\langle x, z_1 \cdots z_p \rangle$, where $x \in \Sigma^*$ and $z_1, \ldots, z_p \in \Sigma^r$, for which there exists a string $w \in \Sigma^p$ having odd parity such that

$$\langle x, y_1, z_1 \rangle \in A_{w_1}, \ldots, \langle x, y_p, z_p \rangle \in A_{w_p}. \tag{2.23}$$

Since A_0 and A_1 are disjoint and contained in $\mathcal{C}$, and $\mathcal{C}$ is closed under joins and truth-table reductions, it follows that $B_0, B_1 \in \mathcal{C}$. Let us illustrate by reducing B_0 to A_1. One way to see the truth-table reduction for B_0 is to consider first a polynomial time computable function that transforms $\langle x, z_1 \cdots z_p \rangle$ to $\langle x, y_1, z_1 \rangle, \ldots, \langle x, y_p, z_p \rangle$. By our choice of an efficient pairing function, we know that such a transformation is possible. Then, we consider the function $f : \Sigma^p \to \Sigma$ defined as

$$f(x_1, \ldots, x_k) = x_1 \oplus \cdots \oplus x_k \oplus 1 \tag{2.24}$$

with $x_1, \ldots, x_k \in \Sigma$. In other words, f computes the complementation of the modulo 2 addition of its input bits. It is easy to see that $\langle x, z_1 \cdots z_p \rangle \in B_0$ if and only if

$$f\big(\chi_{A_1}(\langle x, y_1, z_1 \rangle), \ldots, \chi_{A_1}(\langle x, y_p, z_p \rangle)\big) = 1 \tag{2.25}$$

where χ_{A_1} is the characteristic function for A_1. Hence, B_0 reduces to A_1. We can similarly show that B_1 reduces to A_1 by choosing f as just the modulo 2 addition (without the complementation.)

Observe that

$$g(x) = \Big|\Big\{z \in \Sigma^s \,:\, \langle x, z \rangle \in B_0\Big\}\Big| - \Big|\Big\{z \in \Sigma^s \,:\, \langle x, z \rangle \in B_1\Big\}\Big| \tag{2.26}$$

for all $x \in \Sigma^*$, where $s = p \cdot r$. The lemma follows from (2.26). □

We will now combine the two lemmas stated above to state a new lemma which will be very useful in our proofs.

Lemma 6. *Let $\mathcal{C}$ be a nontrivial complexity class of languages that is closed under joins and polynomial-time truth table reductions, let $f_0, f_1 \in \mathrm{Gap} \cdot \mathcal{C}$, and let p and q be polynomially bounded functions. For every string $x \in \Sigma^*$ and $y \in \Sigma_1^q$, define the matrix $M_{x,y}$ as*

$$\begin{aligned} \mathrm{Re}\big(\langle z | M_{x,y} | w \rangle\big) &= f_0(x, y, z, w), \\ \mathrm{Im}\big(\langle z | M_{x,y} | w \rangle\big) &= f_1(x, y, z, w), \end{aligned} \tag{2.27}$$

for all $z, w \in \Sigma^p$. There exist $\mathrm{Gap} \cdot \mathcal{C}$ functions g_0 and g_1 satisfying

$$\begin{aligned} \mathrm{Re}\big(\langle z | M_{x,y_1} \cdots M_{x,y_q} | w \rangle\big) &= g_0(x, z, w), \\ \mathrm{Im}\big(\langle z | M_{x,y_1} \cdots M_{x,y_q} | w \rangle\big) &= g_1(x, z, w), \end{aligned} \tag{2.28}$$

for all $x \in \Sigma^*$ and $z, w \in \Sigma^p$, where $y_1, \ldots, y_q$ denote the elements of Σ_1^q sorted in lexicographic order.

Proof. Because of the assumptions on $\mathcal{C}$ stated in the lemma, we can argue that there exists a Gap $\cdot\, \mathcal{C}$ function h satisfying

$$\begin{aligned} h(x,y,0z,0w) &= f_0(x,y,z,w), \\ h(x,y,0z,1w) &= f_1(x,y,z,w), \\ h(x,y,1z,0w) &= -f_1(x,y,z,w), \\ h(x,y,1z,1w) &= f_0(x,y,z,w), \end{aligned} \tag{2.29}$$

for all $x \in \Sigma^*$, $y \in \Sigma_1^q$, and $z, w \in \Sigma^p$. Define a matrix $N_{x,y}$ as

$$\langle u|N_{x,y}|v\rangle = h(x,y,u,v) \tag{2.30}$$

for all $x \in \Sigma^*$, $y \in \Sigma_1^q$ and $u, v \in \Sigma^{p+1}$. This matrix can be visualized as a 2×2 block matrix:

$$N_{x,y} = \begin{pmatrix} \operatorname{Re}(M_{x,y}) & \operatorname{Im}(M_{x,y}) \\ -\operatorname{Im}(M_{x,y}) & \operatorname{Re}(M_{x,y}) \end{pmatrix}. \tag{2.31}$$

We can prove by mathematical induction that

$$N_{x,y_1} \cdots N_{x,y_q} = \begin{pmatrix} \operatorname{Re}\big(M_{x,y_1} \cdots M_{x,y_q}\big) & \operatorname{Im}\big(M_{x,y_1} \cdots M_{x,y_q}\big) \\ -\operatorname{Im}\big(M_{x,y_1} \cdots M_{x,y_q}\big) & \operatorname{Re}\big(M_{x,y_1} \cdots M_{x,y_q}\big) \end{pmatrix}. \tag{2.32}$$

Since h is a Gap $\cdot\, \mathcal{C}$ function, we can argue for the existence of a Gap $\cdot\, \mathcal{C}$ function F for which

$$F(x, u_0 \cdots u_q, y_k) = h(x, y_k, u_{k-1}, u_k) \tag{2.33}$$

for all $x \in \Sigma^*$, $u_0, \ldots, u_q \in \Sigma^{p+1}$, and $k \in \{1, \ldots, q\}$.

Let us define

$$G(x, u_0 \cdots u_q) = \prod_{y \in \Sigma_1^q} F(x, u_0 \cdots u_q, y) = h(x, y_1, u_0, u_1) \cdots h(x, y_q, u_{q-1}, u_q) \tag{2.34}$$

for all $x \in \Sigma^*$ and $u_0, \ldots, u_q \in \Sigma^{p+1}$, as well as

$$\begin{aligned} g_0(x,z,w) &= \sum_{u \in \Sigma^{(q-1)(p+1)}} G(x, 0zu0w), \\ g_1(x,z,w) &= \sum_{u \in \Sigma^{(q-1)(p+1)}} G(x, 0zu1w), \end{aligned} \tag{2.35}$$

for all $x \in \Sigma^*$ and $z, w \in \Sigma^p$. It follows by Lemmas 4 and 5 that $g_0, g_1 \in \text{Gap} \cdot \mathcal{C}$.

We observe that g_0 and g_1 satisfy the equations (2.28). A way to see this is to note that

$$G(x, u_0 \cdots u_q) = \langle u_0|N_{x,y_1}|u_1\rangle \cdots \langle u_{q-1}|N_{x,y_q}|u_q\rangle \tag{2.36}$$

and that

$$\sum_{u \in \Sigma^{p+1}} |u\rangle\langle u| = \mathbb{1}_{2^{p+1} \times 2^{p+1}}. \tag{2.37}$$

This means that

$$\begin{aligned} g_0(x,z,w) &= \langle 0z|N_{x,y_1}\cdots N_{x,y_q}|0w\rangle, \\ g_1(x,z,w) &= \langle 0z|N_{x,y_1}\cdots N_{x,y_q}|1w\rangle. \end{aligned} \tag{2.38}$$

The observation is now evident from equation (2.32), and the proof of the lemma is complete. □

Chapter 3

Quantum information

In this chapter, we will give the reader an overview of quantum information. Once again, it is not meant to be exhaustive. The purpose is just to give a clearer account of the notations and conventions used in the rest of the thesis. We refer the reader to books [NC00, KSV02, Wil17, Wat18] for further details. We will begin by defining quantum registers, quantum channels, and quantum circuits. Then, we will elaborate more on the connections between the complexity theoretic tools of the earlier chapter and the concepts discussed here.

3.1. Basics of quantum information

For an alphabet Σ, a *complex Euclidean space* $\mathbb{C}^{\Sigma}$ is the space of all the complex vectors indexed by Σ with addition and scalar multiplication is defined in the following way:

1. *Addition*: For vectors $u, v \in \mathbb{C}^{\Sigma}$, the vector $u + v \in \mathbb{C}^{\Sigma}$ is defined as

$$(u + v)(a) = u(a) + v(a), \tag{3.1}$$

   for all $a \in \Sigma$.

2. *Scalar multiplication*: For a vector $u \in \mathbb{C}^{\Sigma}$, and a scalar $\alpha \in \mathbb{C}$, the vector $\alpha u \in \mathbb{C}^{\Sigma}$ is defined as

$$(\alpha u)(a) = \alpha u(a), \tag{3.2}$$

   for all $a \in \Sigma$.

for all $a \in \Sigma$. The dimension of a complex Euclidean space $\mathbb{C}^{\Sigma}$ is $|\Sigma|$. We will use capital letters to refer to complex Euclidean spaces, like $\mathcal{A}, \mathcal{B}, \mathcal{W}, \mathcal{X}$ etc. We will use the lower case alphabets, like u, v etc, to refer to vectors in a complex Euclidean space. The inner product between two vectors $u, v \in \mathbb{C}^{\Sigma}$ is defined as

$$\langle u, v \rangle = \sum_{a \in \Sigma} \overline{u(a)}\, v(a), \tag{3.3}$$

where $u(a)$ and $v(a)$ refer to the entries indexed by a of vectors u and v respectively. The *Euclidean norm* of a vector $u \in \mathbb{C}^{\Sigma}$ is defined by

$$||u|| = \sqrt{\langle u, u \rangle}. \tag{3.4}$$

We will often refer to probability distributions. For an alphabet Σ, a *probability distribution* $p \in \mathbb{R}^{\Sigma}$ is a vector such that

$$p(a) \geq 0, \tag{3.5}$$

for all $a \in \Sigma$ and

$$\sum_{a \in \Sigma} p(a) = 1. \tag{3.6}$$

The set of all such vectors is denoted by the set $\mathcal{P}(\Sigma)$.

A *register* is a hypothetical device storing quantum information. A register X refers to a collection of qubits we want to view as a single entity. The complex Euclidean space $\mathcal{X}$ is associated with a register X. Each quantum state of the register X is denoted by a density operator $\rho \in \mathrm{D}(\mathcal{X})$. A qubit is a register with $\dim(\mathcal{X}) = 2$. Note that since general registers are collections of qubits, the dimension of the complex Euclidean space corresponding to any register is 2^d for some $d \in \mathbb{N}$. No generality is lost here. This is because, when padded appropriately with zeros, each quantum state of a register with arbitrary dimension is also a quantum state of a register whose dimension is 2 raised to a sufficiently large exponent.

A *channel* transforming a register X into a register Y is a completely positive and trace-preserving linear map Φ that transforms each density operator $\rho \in \mathrm{D}(\mathcal{X})$ into a density operator $\Phi(\rho) \in \mathrm{D}(\mathcal{Y})$. The *adjoint* of Φ is the unique linear map Φ^* transforming linear operators acting on $\mathcal{Y}$ into linear operators acting on $\mathcal{X}$ that satisfies the equation

$$\mathrm{Tr}\big(P\Phi(\rho)\big) = \mathrm{Tr}\big(\Phi^*(P)\rho\big) \tag{3.7}$$

for all density operators $\rho \in \mathrm{D}(\mathcal{X})$ and all positive semidefinite operators P acting on $\mathcal{Y}$. The adjoint map may not be a channel but is a completely positive and *unital* linear map, which means that $\Phi^*(\mathbb{1}_{\mathcal{Y}}) = \mathbb{1}_{\mathcal{X}}$ (for $\mathbb{1}_{\mathcal{X}}$ and $\mathbb{1}_{\mathcal{Y}}$ denoting the identity operators acting on $\mathcal{X}$ and $\mathcal{Y}$, respectively).

A *quantum circuit* is an acyclic network of quantum gates connected by qubit wires. While discussing quantum circuits, we use the general model of quantum information based on density matrices and channels, instead of the restricted model based on unitary operators and pure states. For the general model, each gate is a quantum channel acting on a fixed number of qubits. As noted in [AKN98] or [Wat09], the general model is equivalent to the restricted model. Although no generality is lost by choosing either model, we contend that the general model has many operational advantages:

- Since we consider quantum channels instead of unitary operators, in addition to all the unitary gates of the restricted model, we also have non-unitary gates—like gates that introduce fresh qubits and gates that throw away qubits. This allows us to avoid having to distinguish between input qubits and ancillary qubits, or output

qubits and garbage qubits. The advantage will be clearer in later chapters, especially when we define the different complexity classes.

- Through this model, any classical circuit, including classical circuits that introduce randomness, can be easily viewed as a special case of quantum circuits. We can use our non-unitary gates to introduce the randomness and perform irreversible computations.
- Quantum measurements can be represented by quantum circuits.
- By considering density matrices instead of pure states, we can eliminate the appearance of the irrational number $1/\sqrt{2}$ in many of the formulas that will appear. This is a minor advantage but, for the sake of neatness, it is nevertheless helpful.

We will fix a universal gate set for the remainder of the thesis. The gates in this set include *Hadamard*, *Toffoli*, and *phase-shift* gates (which induce the single-qubit unitary transformation determined by the actions $|0\rangle \mapsto |0\rangle$ and $|1\rangle \mapsto i|1\rangle$), as well as *ancillary gates* and *erasure gates*. Ancillary gates take no input qubits and output a single qubit in the $|0\rangle$ state, while erasure gates take one input qubit and produce no output qubits, and are described by the partial trace. Any other choice of a universal gate set also works just as well. We have chosen this particular one out of simplicity and convenience.

The *size* of a quantum circuit is the total number of gates in the circuit, added to the total number of input and output qubits. If the quantum circuit is represented as a directed acyclic graph with the input and output qubits corresponding to vertices, the size of the circuit is just the number of vertices of the graph.

A collection $\{Q_x : x \in \Sigma^*\}$ of quantum circuits is termed as *polynomial-time generated* if there exists a polynomial-time deterministic Turing machine that, on input $x \in \Sigma^*$, outputs an encoding of the circuit Q_x. As is clear from the definition, the size of a polynomial-time generated quantum circuit is always upper-bounded by a polynomially bounded function in the input length. When such a family is parameterized by tuples of strings, it is implicit that we are referring to one of the tuple-functions discussed previously. We will not get into the details of the encoding function and will just note that the encoding should be efficient, with the size of the circuit polynomially related to its encoding length.

3.2. Relation to complexity theory

We will state a lemma that relates quantum circuits to GapP functions discussed in Chapter 2. Fortnow and Rogers [FR99] proved a variant of the lemma for quantum Turing machines. A similar result is also discussed in [Wat09] without a formal proof. We include a comprehensive proof below. A result that is similar in spirit, relating quantum circuits to counting classes, can be found in [DHM+05].

Lemma 7. *Let $\{Q_x : x \in \Sigma^*\}$ be a polynomial-time generated family of quantum circuits, where each circuit Q_x takes n input qubits and outputs k qubits, for polynomially bounded functions n and k. There exists a polynomially bounded function r and* GapP *functions f_0 and f_1 such that*

$$\begin{aligned} \mathrm{Re}\big(\langle u|Q_x\big(|z\rangle\langle w|\big)|v\rangle\big) &= 2^{-r} f_0(x,z,w,u,v), \\ \mathrm{Im}\big(\langle u|Q_x\big(|z\rangle\langle w|\big)|v\rangle\big) &= 2^{-r} f_1(x,z,w,u,v), \end{aligned} \tag{3.8}$$

for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u, v \in \Sigma^k$.

Proof. Let us begin by considering first an arbitrary channel Φ that maps n-qubit density operators to k-qubit density operators. Since the action of Φ on density operators is linear, it can therefore be represented by means of matrix multiplication. A way to do this is by the *natural representation* (also known as the linear representation) of quantum channels. We give a brief description below. It starts with a description of the *vectorization* mapping.

Assume that M is a matrix whose rows and columns are indexed by strings of some length m. The corresponding vector $\mathrm{vec}(M)$ is defined as

$$\mathrm{vec}(M) = \sum_{y,z \in \Sigma^m} \langle y|M|z\rangle \, |yz\rangle. \tag{3.9}$$

Observe that it is indexed by strings of length $2m$. Qualitatively, the vectorization map makes a vector out of a matrix by transposing the rows of the matrix into column vectors and putting them on top of one another.

The natural representation $K(\Phi)$ of a channel Φ is defined as the linear mapping that has the following action:

$$K(\Phi)\,\mathrm{vec}(\rho) = \mathrm{vec}(\Phi(\rho)) \tag{3.10}$$

for every n-qubit density operator ρ.

The matrix entries of $K(\Phi)$ are explicitly given by the equation

$$\langle uv|K(\Phi)|zw\rangle = \langle u|\Phi(|z\rangle\langle w|)|v\rangle \tag{3.11}$$

holding for every $z, w \in \Sigma^n$ and $u, v \in \Sigma^k$. Note that the matrix $K(\Phi)$ has columns indexed by strings of length $2n$ and rows indexed by strings of length $2k$. Thus, the equations (3.8) are equivalent to the equations

$$\begin{aligned} \mathrm{Re}\big(\langle uv|K(Q_x)|zw\rangle\big) &= 2^{-r} f_0(x,z,w,u,v), \\ \mathrm{Im}\big(\langle uv|K(Q_x)|zw\rangle\big) &= 2^{-r} f_1(x,z,w,u,v). \end{aligned} \tag{3.12}$$

Note that the natural representation is multiplicative. In other words,

$$K(\Phi\Psi) = K(\Phi)K(\Psi) \tag{3.13}$$

for all channels Φ and Ψ for which the composition $\Phi\Psi$ makes sense. Also note that a channel $\Phi(\rho) = U\rho U^*$ which corresponds to a unitary operation U has as its natural representation the operator

$$K(\Phi) = U \otimes \overline{U}. \tag{3.14}$$

Let us consider the circuit family $\{Q_x : x \in \Sigma^*\}$. Since this family is polynomial-time generated, from our discussion about polynomial-time generated circuits in the earlier section, there must exist a polynomially bounded function r such that $\text{size}(Q_x) \leq r$ for all $x \in \Sigma^*$. We may therefore write

$$Q_x = Q_{x,r} \cdots Q_{x,1}, \tag{3.15}$$

where $Q_{x,1}, \ldots, Q_{x,r}$ are either identity channels or channels that correspond to the action of a single gate of Q_x tensored with the identity channel on all of the qubits other than the qubits that are input to that single gate. It is easy to see that the number of input qubits and output qubits of each $Q_{x,k}$ must be bounded by r.

Since we have

$$K(Q_x) = K(Q_{x,r}) \cdots K(Q_{x,1}), \tag{3.16}$$

we will consider the natural representation of each channel $Q_{x,k}$. For convenience, let us identify each operator $K(Q_{x,k})$ with the matrix indexed by strings of length $2r$, instead of being indexed by strings whose lengths depend on the number of qubits in existence before and after $Q_{x,k}$ is applied. We can achieve this simply by padding $K(Q_{x,k})$ with rows and columns of zero entries.

The natural representations of the individual gates in the universal gate set we have defined in the previous section are as follows:

1. Hadamard gate:

$$\frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \tag{3.17}$$

2. Phase gate:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -i & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.18}$$

3. Toffoli gate:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{3.19}$$

4. Ancillary qubit gate:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.20}$$

5. Erasure gate:

$$\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} \tag{3.21}$$

On the basis of these representations, we can see that the real and imaginary parts of the matrix entries of each of these gates come from the set $\{-1, -1/2, 0, 1/2, 1\}$. The $-1/2$ and $1/2$ entries of the set come from the Hadamard gate. Keeping this in mind (and the fact that Q_x is polynomial-time generated), it is straightforward to argue for the existence of GapP functions (or, in fact, FP functions [1]) g_0 and g_1 such that

$$\begin{aligned} \mathrm{Re}\big(\langle uv|K(Q_{x,k})|zw\rangle\big) &= \frac{1}{2}\, g_0(x, z, w, u, v, y_k), \\ \mathrm{Im}\big(\langle uv|K(Q_{x,k})|zw\rangle\big) &= \frac{1}{2}\, g_1(x, z, w, u, v, y_k), \end{aligned} \tag{3.22}$$

for all $x \in \Sigma^*$, $k \in \{1, \ldots, r\}$, and $u, v, z, w \in \Sigma^r$, where $y_1, \ldots, y_r$ denote the elements of Σ_1^r sorted in lexicographic order. The factor of $1/2$ in front of g_0 and g_1 comes from the Hadamard gate. Intuitively, the fact that g_0 and g_1 are in FP can be visualized by noting that there exists a polynomial-time deterministic Turing machine such that it takes as input $\langle x, z, w, u, v, y_k\rangle$, computes an encoding of Q_x in polynomial time (it can do so as Q_x is polynomial time generated), efficiently figures out what $Q_{x,k}$ is from y_k (since we assume efficient pairing-functions, y_k is efficiently computable from $\langle x, z, w, u, v, y_k\rangle$), efficiently figures out u, v, z, and w from $\langle x, z, w, u, v, y_k\rangle$ (once again, it can do so because of our assumption of an efficient pairing function for the input), and outputs twice the real or imaginary part (depending on g_0 or g_1) of the $(u, v, z, w)^{\text{th}}$ entry of $K(Q_{x,k})$ (we consider a value that is twice the actual entry of $K(Q_{x,k})$ to adjust for the $1/2$ in front of the natural representation of the Hadamard gate and make the output an integer for any input). Note that $Q_{x,k}$ is one of the five universal gates, padded with zeroes. The zero padding depends on r but is independent of the input to the machine. We can assume that the matrices corresponding to twice the natural representation of these five gates, with requisite zero padding, are hard-coded into the memory of the machine from the start. Since the entries of these matrices are independent of the input, accessing them from the memory tape incurs only a constant overhead. Observe that the whole operation of this machine takes time that is polynomial in the length of the input. Also observe that depending on whether it outputs the real or the imaginary part, this machine computes g_0 or g_1, which puts them in FP.

[1] FP functions are a class of functions $f : \Sigma^* \to \mathbb{Z}$ that are computable in polynomial time, with the integer output being encoded in binary. It can be easily proven that FP is contained in GapP.

It now follows through a direct application of Lemma 6 there must exist GapP functions f_0 and f_1 satisfying (3.8) and therefore (3.12), for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u, v \in \Sigma^k$, thus completing the proof. □

Chapter 4

Definitions

This chapter will serve as a warm-up to our results. Here, we will rigorously define all the unfamiliar complexity classes, both quantum and classical, that we will consider for the rest of the thesis. For familiar complexity classes, like QMA and PP, we refer the reader to the survey by Watrous [Wat09]. For the quantum classes, we will start with the definition of QRG(1). Then, we will define CQRG(1) and MQRG(1). For the classical cases, we will start with a brief recap of $\exists \cdot \mathrm{PP}$ and $\mathrm{P} \cdot \mathrm{PP}$ from Chapter 2.

4.1. Quantum complexity classes

In this section, we define the three quantum complexity classes to be considered in this thesis: $\mathrm{QRG}(1)$, $\mathrm{CQRG}(1)$, and $\mathrm{MQRG}(1)$. While defining all these quantum classes, we will always talk about a *referee*. Hence, before we start, we will formalize the notion of the same.

Definition 8. A referee is a polynomial-time generated family

$$R = \{R_x \,:\, x \in \Sigma^*\} \tag{4.1}$$

of quantum circuits which has the following features, for each $x \in \Sigma^*$:

1. The inputs to the circuit R_x can be divided into two registers: an n-qubit register A and an m-qubit register B, where n and m are polynomially bounded functions.
2. The output of the circuit R_x is a single qubit, which is measured in the standard basis immediately after running the circuit.

This definition of the referee works just as well when one of the inputs to the referee is a classical state. This is because any classical probability distribution, which represents a general classical state, can be represented in turn as a quantum state, by a diagonal density matrix.

In our definitions that follow, there will be two players, Alice and Bob, sending two quantum (or classical) states to the referee, for a particular input string $x \in \Sigma^*$. The input

to the circuit R_x is of the form $\rho \otimes \sigma$, where $\rho \in \mathrm{D}(\mathcal{A})$ is sent by Alice, and $\sigma \in \mathrm{D}(\mathcal{B})$ is sent by Bob. The state ρ is stored in register A and σ is stored in register B. When the measurement of the output qubit of R_x in the standard basis yields outcome 1, we interpret it as "Alice wins." Similarly, if it yields outcome 0, we interpret it as "Bob wins."

Now, let us consider the quantity $\omega(R_x)$ defined below.

$$\omega(R_x) = \max_{\rho \in \mathrm{D}(\mathcal{A})} \min_{\sigma \in \mathrm{D}(\mathcal{B})} \langle 1|R_x(\rho \otimes \sigma)|1\rangle. \tag{4.2}$$

Observe that $\mathrm{D}(\mathcal{A})$ and $\mathrm{D}(\mathcal{B})$ are compact and convex sets, and the value $\langle 1|R_x(\rho \otimes \sigma)|1\rangle$ is bilinear in ρ and σ. Applying Sion's min-max theorem, we can argue that changing the order of the minimum and maximum does not change the value of the expression. In other words, this quantity may alternatively be written

$$\omega(R_x) = \min_{\sigma \in \mathrm{D}(\mathcal{B})} \max_{\rho \in \mathrm{D}(\mathcal{A})} \langle 1|R_x(\rho \otimes \sigma)|1\rangle. \tag{4.3}$$

Note that this value represents the probability that Alice "wins the game" when the referee's circuit is described by R_x, for a particular $x \in \Sigma^*$, assuming both Alice and Bob play optimally. With this quantity defined, let us now turn our attention to the definition of QRG(1).

Definition 9. A promise problem $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ is contained in the complexity class $\mathrm{QRG}(1)_{\alpha,\beta}$ if there exists a referee $R = \{R_x : x \in \Sigma^*\}$ such that the following properties are satisfied:

1. For every string $x \in A_{\mathrm{yes}}$, it is the case that $\omega(R_x) \geq \alpha$.
2. For every string $x \in A_{\mathrm{no}}$, it is the case that $\omega(R_x) \leq \beta$.

We also define $\mathrm{QRG}(1) = \mathrm{QRG}(1)_{2/3,1/3}$.

In Definition 9, α and β may be constants, or they may be functions of the length of the input x. A few known facts about QRG(1) are summarized below.

- QMA $\subseteq$ QRG(1). The QRG(1) referee may discard Bob's state and only consider Alice's state ρ as a quantum proof. Thus, any QMA referee has an analogous QRG(1) referee.
- QRG(1) is closed under complementation: QRG(1) = co-QRG(1). For a promise problem $A = (A_{\mathrm{yes}}, A_{\mathrm{no}}) \in \mathrm{QRG}(1)$, we can just exchange the roles of Alice and Bob to obtain a new one-turn quantum refereed game for A.
- It is true that, like the error bounds for BPP, BQP, and QMA, $\mathrm{QRG}(1) = \mathrm{QRG}(1)_{\alpha,\beta}$ if α and β are polynomial-time computable functions and satisfy the following relations:

  $$\alpha \leq 1 - 2^{-p}, \quad \beta \geq 2^{-p}, \quad \text{and} \quad \alpha - \beta \geq \frac{1}{p} \tag{4.4}$$

  for some choice of a strictly positive polynomially bounded function p.

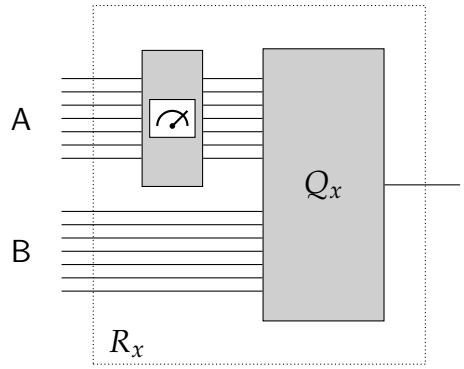

Figure 4.1: A pictorial depiction of a CQRG(1) referee. The register A is initially measured (or, in more technical terms, dephased) with respect to the standard basis, which results in a classical state to be input into Q_x, along with the register B, which is not disturbed by this standard basis measurement.

- Error reduction in QRG(1) can be performed by means of parallel repetition followed by majority vote. An analysis of the parallel repetition method in QRG(1) requires one to take into account the fact that the dishonest player—the player who is supposed to lose with high probability—can try to cheat by sending a state that is entangled across the different repetitions of the game, instead of sending a product state. The analysis of this method is similar in spirit to the analysis of parallel repetition followed by majority vote for QMA [KSV02]. Note that unlike the Marriott-Watrous technique of QMA [MW05], no "in place" error reduction scheme is known for QRG(1).
- $\mathrm{QRG}(1) \subseteq \mathrm{PSPACE}$ [JW09].

Originally, we set out to prove a better containment than PSPACE for QRG(1). We did not succeed in proving that containment. However, we proved better containments for two restricted versions of QRG(1). The first such complexity class is called CQRG(1): in $\mathrm{CQRG}(1)$, Alice's state $\rho \in \mathrm{D}(\mathcal{A})$ is restricted to be a classical state.

Definition 10. A promise problem $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ is contained in the complexity class $\mathrm{CQRG}(1)_{\alpha,\beta}$ if there exists a referee $R = \{R_x : x \in \Sigma^*\}$ such that the following properties are satisfied:

1. For every string $x \in \Sigma^*$, the circuit R_x takes the form depicted in Figure 4.1. In other words, R_x takes an n-qubit register A and an m-qubit register B as input, measures each qubit of A with respect to the standard basis, leaving it in a classical state, and then runs the circuit Q_x on the pair (A, B), producing a single output qubit.
2. For every string $x \in A_{\mathrm{yes}}$, it is the case that $\omega(R_x) \geq \alpha$.
3. For every string $x \in A_{\mathrm{no}}$, it is the case that $\omega(R_x) \leq \beta$.

Like in the previous definition, we define $\mathrm{CQRG}(1) = \mathrm{CQRG}(1)_{2/3,1/3}$.

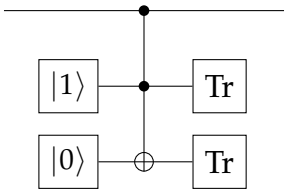

Figure 4.2: A pictorial representation of a dephasing channel made of universal gates.

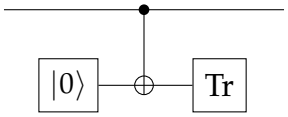

Figure 4.3: A second pictorial representation of a dephasing channel made of a different set of universal gates.

In more technical terms, the standard basis measurement in Definition 10 can be implemented by independently applying the completely dephasing channel on each qubit of A. The completely dephasing channel can be constructed with the universal gate set we defined in Chapter 3. For the sake of completeness, we give a description below. In Figure 4.2, the square labeled $|0\rangle$ is an ancillary gate, the square labeled $|1\rangle$ is an ancillary gate composed with a not-gate $X = HPPH$, where H and P denote Hadamard and phase-shift gates, and the square labeled Tr denotes an erasure gate. A way to see that the circuit above performs the completely dephasing channel is be to consider the density matrix of a general quantum state on the first qubit, consider the density matrix of all three qubits by tensoring the density matrix of the ancillas with that of the first qubit, apply the Toffoli gate to the density matrix over three qubits, and then consider the reduced density matrix of the first qubit after tracing out the second and third qubit. We will see that the reduced density matrix of the first qubit after the final step contains only the diagonal entries.

Note that there is nothing unique or special about the universal gate set we defined or the way we constructed the completely dephasing channel. If we fixed a different universal gate set, we would get a different way to construct the channel. For example, if we chose *Hadamard*, *controlled NOT*, *phase shift*, and the T gate in our universal gate set, along with ancillary and erasure gates, where the T gate is given as

$$\begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}, \tag{4.5}$$

the dephasing channel can be constructed as in Figure 4.3.

Qualitatively, a referee R that meets the first requirement of Definition 10 forces Alice to be classical (i.e., play a state represented by a diagonal density operator). Mathematically, for any density operator ρ that Alice chooses to play, the state of A that goes into Q_x

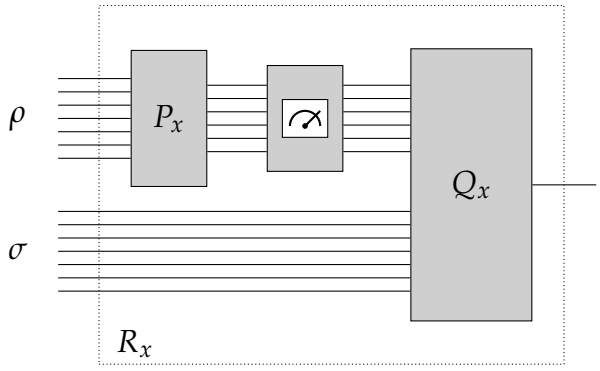

Figure 4.4: An MQRG(1) referee.

is given by

$$\sum_{y \in \Sigma^n} p(y) \, |y\rangle\langle y| \tag{4.6}$$

for some probability vector p over n-bit strings. In other words, the first n qubits of Q_x represent a diagonal density operator (i.e., a classical state). Since that the standard basis measurement acts trivially on all diagonal density operators, instead of sending a quantum state which then gets dephased and turned to a diagonal density state, Alice may just start with an arbitrary diagonal density state of the form (4.6) from the very beginning. To summarize, an analysis of Figure 4.1 reveals that the set of possible states that may be input into the first n qubits of the circuit Q_x is precisely the set of n-qubit diagonal density operators.

Now, we will define a second restricted version of QRG(1). In this class, Alice and Bob both send quantum states to the referee, but the referee measures Alice's state first, obtains a classical outcome, which is then measured together with Bob's state. We call this class MQRG(1). The referee's action is illustrated in Figure 4.4.

Definition 11. A promise problem $A = (A_{\text{yes}}, A_{\text{no}})$ is contained in the complexity class $\text{MQRG}(1)_{\alpha,\beta}$ if there exists a referee $R = \{R_x : x \in \Sigma^*\}$ such that the following properties are satisfied:

1. For each string $x \in \Sigma^*$, the circuit R_x takes the form of Figure 4.4. To elaborate, R_x takes an n-qubit register A and an m-qubit register B as input, and first applies a quantum circuit P_x to A, producing a k-qubit register Y, where k is a polynomially bounded function. Then, the register Y is measured with respect to the standard basis, so that it then stores a classical state. Finally, a quantum circuit Q_x is applied to the pair (Y, B), producing a single qubit.
2. For every string $x \in A_{\text{yes}}$, it is the case that $\omega(R_x) \geq \alpha$.
3. For every string $x \in A_{\text{no}}$, it is the case that $\omega(R_x) \leq \beta$.

As before, we also define $\text{MQRG}(1) = \text{MQRG}(1)_{2/3,1/3}$.

Qualitatively, an MQRG(1) referee measures Alice's qubits with respect to a general measurement that we can implement efficiently and produces a k-bit classical state, which is then plugged into Q_x alongside Bob's quantum state.

Note that a CQRG(1) referee is a special case of an MQRG(1) referee in which P_x is the identity map on n qubits. In turn, MQRG(1) referee is a special case of a QRG(1) referee. It follows immediately that

$$\text{CQRG}(1) \subseteq \text{MQRG}(1) \subseteq \text{QRG}(1). \tag{4.7}$$

Remark 12. *Note that when P_x is restricted to be a unitary quantum circuit, the containment* MQRG(1) $\subseteq$ CQRG(1) *also holds, which implies that for this special case,* CQRG(1) $=$ MQRG(1). *A way to see the containment is to note that since P_x is assumed to be a unitary quantum circuit, for every state $\rho \in \mathrm{D}(\mathcal{A})$ that Alice sends to an* MQRG(1) *referee, she can send the diagonal density matrix $\Delta^{\otimes n}(P_x \rho P_x^*) \in \mathrm{D}(\mathcal{A})$ to a* CQRG(1) *referee, where $\Delta^{\otimes n}$ is the n-fold tensor product of the one-qubit completely dephasing channel Δ, which when applied to the density matrix of a qubit leaves the diagonal entries intact and sets the off-diagonal entries to zero.*

Observe that both CQRG(1) and MQRG(1) are robust with respect to error bounds, similar to QRG(1).

4.2. Classical complexity classes

In this section, we define the classical complexity classes we consider in this thesis: $\exists \cdot \mathrm{PP}$ and $\mathrm{P} \cdot \mathrm{PP}$. These classes are all contained in PSPACE. These were already defined in general terms in Chapter 2. We give a brief recap for convenience. We also include a short proof that they are contained in PSPACE.

Definition 13. The complexity class $\exists \cdot \mathrm{PP}$ contains all promise problems $A = (A_{\text{yes}}, A_{\text{no}})$ for which there exists a language $B \in \mathrm{PP}$ and a polynomially bounded function p such that these two implications hold:

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} \neq \varnothing, \\ x \in A_{\text{no}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} = \varnothing. \end{aligned} \tag{4.8}$$

Definition 14. The complexity class $\mathrm{P} \cdot \mathrm{PP}$ contains all promise problems $A = (A_{\text{yes}}, A_{\text{no}})$ for which there exists a language $B \in \mathrm{PP}$ and a polynomially bounded function p such that these two implications hold:

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left| \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} \right| > \frac{1}{2} \cdot 2^p, \\ x \in A_{\text{no}} &\Rightarrow \left| \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} \right| \leq \frac{1}{2} \cdot 2^p. \end{aligned} \tag{4.9}$$

Proposition 15. $\exists \cdot \mathrm{PP} \subseteq \mathrm{P} \cdot \mathrm{PP}$

Proof. Consider a promise problem A in $\exists \cdot \mathrm{PP}$ such that there exists a language $B \in \mathrm{PP}$ and a polynomially bounded function p for which:

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} \neq \varnothing, \\ x \in A_{\text{no}} &\Rightarrow \left\{ y \in \Sigma^p \,:\, \langle x, y \rangle \in B \right\} = \varnothing. \end{aligned} \tag{4.10}$$

Consider a language C defined as

$$C = \{x0 : x \in \Sigma^*\} \cup \{x1 : x \in B\}. \tag{4.11}$$

Since PP is closed under complementation, joins, and unions, it follows that $C \in \mathrm{PP}$. It is now easy to see that $p + 1$ is a polynomially bounded function such that

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left| \left\{ y \in \Sigma^{p+1} \,:\, \langle x, y \rangle \in C \right\} \right| > \frac{1}{2} \cdot 2^{p+1}, \\ x \in A_{\text{no}} &\Rightarrow \left| \left\{ y \in \Sigma^{p+1} \,:\, \langle x, y \rangle \in C \right\} \right| \leq \frac{1}{2} \cdot 2^{p+1}. \end{aligned} \tag{4.12}$$

Since Definition 14 is satisfied, the lemma follows. □

Proposition 16. $\mathrm{P} \cdot \mathrm{PP} \subseteq \mathrm{PSPACE}$

Proof. Consider a promise problem $A \in \mathrm{P} \cdot \mathrm{PP}$. From the definition, there exists a language $B \in \mathrm{PP}$ and a polynomially bounded function p such that

$$\begin{aligned} x \in A_{\text{yes}} &\Rightarrow \left| \left\{ y \in \Sigma^{p} \,:\, \langle x, y \rangle \in B \right\} \right| > \frac{1}{2} \cdot 2^{p}, \\ x \in A_{\text{no}} &\Rightarrow \left| \left\{ y \in \Sigma^{p} \,:\, \langle x, y \rangle \in B \right\} \right| \leq \frac{1}{2} \cdot 2^{p}. \end{aligned} \tag{4.13}$$

Since $\mathrm{PP} \subseteq \mathrm{PSPACE}$, we have that $B \in \mathrm{PSPACE}$. Let N be a PSPACE machine that establishes this containment. Consider a machine M taking $x \in \Sigma^*$ as input. The action of M is described as follows:

- M iterates over every $y \in \Sigma^p$.
- During each iteration, M simulates the action of N on input $\langle x, y \rangle$.
- M has a p-bit register, initialized to all 0s, that it increments everytime N accepts.
- At the end of each iteration, M stores nothing except for the contents of that p-bit register, and reuses memory space during each new iteration.
- M stops and accepts whenever the binary string stored in the p-bit register is more than $\frac{1}{2} \cdot 2^p$.

- M rejects if, even after iterating over every $y \in \Sigma^p$, the binary string stored in the p-bit register is less than or equal to $\frac{1}{2} \cdot 2^p$.

Note that since M reuses space during every iteration, it takes, at most, a polynomial amount of space. Also note that M accepts every string $x \in A_{\text{yes}}$ and rejects every string $x \in A_{\text{no}}$. For any x in the promise gap, since N outputs garbage for any $y \in \Sigma^p$, M also outputs garbage at the end. It is evident that, with respect to the machine M, A is contained in PSPACE. □

Chapter 5

Results - Part I

In this chapter, we will first state a folklore result: $\mathrm{QRG}(1) = \mathrm{P}^{\mathrm{QRG}(1)}$. Even though this result is known, there has not been a detailed proof of it in literature. We will give a proof here. Then, we will elaborate on a Chernoff-type bound on matrix valued random variables that will be used to prove both of our main containment results. The bound is due to a more general result by [Tro12]. Finally, we will explain one of our main containment result: $\mathrm{CQRG}(1) \subseteq \exists \cdot \mathrm{PP}$. The other containment result will be proven in the next chapter.

5.1. Folklore result

Here, we will prove that the complexity class QRG(1) is equal to the complexity class $\mathrm{P}^{\mathrm{QRG}(1)}$. We will split the containment into two parts, first proving that $\mathrm{QRG}(1) \subseteq \mathrm{P}^{\mathrm{QRG}(1)}$ and then proving that $\mathrm{P}^{\mathrm{QRG}(1)} \subseteq \mathrm{QRG}(1)$. The second containment only holds if, for each $x \in \Sigma^*$, each query made by the $\mathrm{P}^{\mathrm{QRG}(1)}$ machine to the QRG(1) oracle lies within the promise. In other words, for a particular x, each query is a string $y \in B_{\mathrm{yes}} \cup B_{\mathrm{no}}$ for a promise problem $B \in \mathrm{QRG}(1)$. [1]

Theorem 17. $\mathrm{QRG}(1) = \mathrm{P}^{\mathrm{QRG}(1)}$.

Proof. It is easy to prove that $\mathrm{QRG}(1) \subseteq \mathrm{P}^{\mathrm{QRG}(1)}$. Consider a promise problem $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ in QRG(1). To construct a $\mathrm{P}^{\mathrm{QRG}(1)}$ algorithm to decide A, just query the QRG(1) oracle and accept or reject based on the answer.

Proving $\mathrm{P}^{\mathrm{QRG}(1)} \subseteq \mathrm{QRG}(1)$ is slightly more involved. Consider a promise problem $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ in $\mathrm{P}^{\mathrm{QRG}(1)}$. For every input $x \in \Sigma^*$, we can assume, without loss of generality, that the polynomial-time algorithm that decides A makes exactly $p(|x|)$ queries to the QRG(1) oracle, for a polynomially bounded function p. The algorithm can be visualized as a complete binary decision tree. Each node of the tree represents a query to the QRG(1) oracle. Depending on whether the oracle answers 0 or 1, there are two edges from each node, representing two different polynomial time computation paths. The leaves of

[1] Reductions where querying the promise gap is forbidden are called "smart" reductions. See Goldreich [Gol06] (especially section 5.1) for a primer.

the tree represent accept and reject nodes. Since it is a complete binary tree, the leaves are all at the same level. Depending on the input x, the algorithm traverses the tree deterministically and terminates at an accept or a reject node.

Now, consider a QRG(1) referee for A and consider an input x. Alice, the yes player, wants to convince the referee that the input leads to an accept node. Bob, the no player, wants to convince the referee that the input leads to a reject node. Let us look at the strategies of Alice and Bob below.

- **Alice's strategy**: Alice sends a binary string $a_1 a_2 \cdots a_p \in \Sigma^p$ and registers

$$(\mathsf{A}_1, \mathsf{A}_2, \ldots, \mathsf{A}_\mathsf{p}), \tag{5.1}$$

  each n qubits long. The register A_i stores a quantum proof that the bit a_i is indeed the correct answer to the i^{th} query for $i \in \{1, 2, \ldots, p\}$. Let the quantum state across the p registers be $\rho \in \mathrm{D}(\mathcal{A}^{\otimes p})$.

- **Bob's strategy**: Bob sends a binary string $b_1 b_2 \cdots b_p \in \Sigma^p$ and registers

$$(\mathsf{B}_1, \mathsf{B}_2, \ldots, \mathsf{B}_\mathsf{p}), \tag{5.2}$$

  each m qubits long. The register B_i stores a quantum proof that the bit b_i is indeed the correct answer to the i^{th} query for $i \in \{1, 2, \ldots, p\}$. Let the quantum state across the p registers be $\sigma \in \mathrm{D}(\mathcal{B}^{\otimes p})$.

- **Referee's action**: The referee performs three tasks:

  - Firstly, the referee actually verifies the computation. He considers Alice's string $a_1 a_2 \ldots a_p$, considers the polynomial-time computation path through the decision tree where the answer to the i^{th} query is indeed a_i, for $i \in \{1, 2, \ldots, p\}$, and verifies whether that polynomial-time computation path indeed leads to an accept node. If it leads to a reject node instead, he declares Bob as the winner by outputting 0.
  - If it does lead to an accept node, he considers Bob's string $b_1 b_2 \ldots b_p$ and sees whether there is any mismatch between Alice's string and Bob's string. If there is none, he declares Alice as the winner (as Bob's string would have led him to the same accept node as that of Alice's string, instead of a reject node) by outputting 1.
  - If there is a mismatch, he considers the first such position $j \in \{1, 2, \ldots, p\}$ that Alice's and Bob's strings disagree on. He considers the quantum circuit which simulates the action of the oracle for the j^{th} position. Note that if $a_j = 0$, Alice is the no-player and Bob is the yes-player, with respect to the j^{th} oracle query. Similarly, If $a_j = 1$, Alice is the yes-player and Bob is the no-player, with respect to the j^{th} oracle query. With this subtle distinction in mind, he plugs in the reduced density matrices $\rho[\mathsf{A}_j] \in \mathrm{D}(\mathcal{A})$——which is the quantum state of the

register A_j with all the other registers traced out—and $\sigma[\mathsf{B}_j] \in \mathrm{D}(\mathcal{B})$——which is the quantum state of the register B_j with all the other registers traced out—— into his quantum circuit. Note that depending on who is the yes-player and who is the no-player, the proofs are plugged into the circuit in their proper places. The referee outputs 1, signaling Alice has won, if the output of the circuit is a_j and outputs 0, signaling Bob has won, if it is b_j.

Let us analyze the referee's action. Observe that if $x \in A_{\text{yes}}$, Alice is the honest player and a_j, the j^{th} entry of the string $a_1 a_2 \cdots a_p$, represents the correct answer of the oracle to the j^{th} query, for every $j \in [p]$. Hence, it follows from the definition of QRG(1) that Alice can send a state $\rho_1 \otimes \rho_2 \otimes \cdots \otimes \rho_p \in \mathrm{D}(\mathcal{A}^{\otimes p})$ such that $\rho_j \in \mathrm{D}(\mathcal{A})$ allows her to win with probability at least $2/3$ for the j^{th} query, for every $j \in [p]$, regardless of whether she is the yes-prover or the no-prover for that instance, and regardless of any $\sigma[\mathsf{B}_j] \in \mathrm{D}(\mathcal{B})$, even if $\sigma \in \mathrm{D}(\mathcal{B}^{\otimes p})$ does not take a product state form. Informally, Alice wins with high probability by sending a product state irrespective of whether Bob tries to cheat by entangling his state across all the p registers.

Similarly, if $x \in A_{\text{no}}$, Bob is the honest player and b_j, the j^{th} entry of the string $b_1 b_2 \cdots b_p$, represents the correct answer of the oracle to the j^{th} query, for every $j \in [p]$. Hence, it follows from the definition of QRG(1) that Bob can send a state $\sigma_1 \otimes \sigma_2 \otimes \cdots \otimes \sigma_p \in \mathrm{D}(\mathcal{B}^{\otimes p})$ such that $\sigma_j \in \mathrm{D}(\mathcal{A})$ allows him to win with probability at least $2/3$ for the j^{th} query, for every $j \in [p]$, regardless of whether he is the yes-prover or the no-prover for that instance, and regardless of any $\rho[\mathsf{A}_j] \in \mathrm{D}(\mathcal{A})$, even if $\rho \in \mathrm{D}(\mathcal{A}^{\otimes p})$ does not take a product state form. Informally, Bob wins with high probability by sending a product state, irrespective of whether Alice tries to cheat by entangling her state across all the p registers.

Note that notwithstanding whether there is any mismatch in Alice's or Bob's strings and whether Alice is the yes-prover or the no-prover for the j^{th} instance, the referee outputs 1 if Alice wins and 0 if Bob wins. This fact, along with our discussion above, implies that when $x \in A_{\text{yes}}$, the referee outputs 1 with probability at least $2/3$ and when $x \in A_{\text{no}}$, the referee outputs 1 with probability at most $1/3$. This proves that $\mathrm{P}^{\mathrm{QRG}(1)} \subseteq \mathrm{QRG}(1)$. Having established both the directions, the proof of the theorem is complete. □

5.2. A tail bound for operator-valued random variables

Now, we will state a tail bound for operator valued random variables that we will use to upper bound CQRG(1). Let us first look at what operator-valued random variables are.

Definition 18. For a given alphabet Σ and a given probability distribution $p \in \mathcal{P}(\Sigma)$, an operator-valued random variable X, distributed with respect to p, is a function of the form

$$X : \Sigma \to \mathrm{L}(\mathcal{Y}, \mathcal{Z}), \tag{5.3}$$

for some fixed choice of complex Euclidean spaces $\mathcal{Y}$ and $\mathcal{Z}$, such that, for every subset $\mathcal{T} \subseteq \Sigma$, the probability that X takes a value in $\mathcal{T}$ is defined as

$$\Pr(X \in \mathcal{T}) = \sum_{a \in \mathcal{T}} p(a). \tag{5.4}$$

The *expected value* of such a random variable is

$$\mathrm{E}(X) = \sum_{a \in \Sigma} p(a) X(a). \tag{5.5}$$

Now, the bound stated in the corollary below follows from Theorem 5.1 of [Tro12] together with Pinsker's inequality, which relates the relative entropy of two probability vectors to the 1-norm of the two vectors. A simplified form of Pinsker's inequality—the form which we use in our proof—follows.

Theorem 19 (Pinsker's inequality)**.** *Let P and Q be two probability distributions over the alphabet Σ with $P(0) = p$, $P(1) = 1 - p$, $Q(0) = q$, and $Q(1) = 1 - q$. Let*

$$D(P||Q) = p \log \frac{p}{q} + (1 - p) \log \frac{1 - p}{1 - q} \tag{5.6}$$

be defined as the relative entropy between the two distributions and

$$||P - Q||_1 = 2|p - q| \tag{5.7}$$

be defined as the L_1 norm between the two distributions. Then, we have

$$D(P||Q) \geq \frac{1}{2 \ln 2} ||P - Q||_1^2. \tag{5.8}$$

Next, we state the tail bound. The proof directly follows from Theorem 5.1 of [Tro12]. We will state a simplified version of this theorem and then state our required tail bound as a corollary. We will also include a short proof of the corollary, just for the sake of completeness.

Theorem 20 (Tropp)**.** *Let $\eta, \alpha \in [0, 1]$ with $\eta \geq \alpha$. Let $X_1, \ldots, X_N$ be independent operator-valued random variables having the following properties:*

1. *Each X_k takes $d \times d$ positive semidefinite operator values satisfying $X_k \leq \mathbb{1}$.*
2. *The minimum eigenvalue of the average of the expectation value of each of the random variables satisfies*

$$\lambda_{\min} \left(\frac{1}{N} \sum_{k=1}^{N} \mathrm{E}(X_k) \right) \geq \eta. \tag{5.9}$$

It is the case that

$$\Pr\left(\lambda_{\min}\left(\frac{X_1+\cdots+X_N}{N}\right)<\alpha\right)\leq d\exp(-N\cdot D(\alpha||\eta)), \tag{5.10}$$

where the relative entropy $D(\alpha||\eta))$ for scalars α and η is defined as

$$D(\alpha||\eta)) = \alpha\log\frac{\alpha}{\eta}+(1-\alpha)\log\frac{1-\alpha}{1-\eta}. \tag{5.11}$$

Remark 21. *Originally, in the statement of this theorem, the author had that*

$$\lambda_{\min}\left(\frac{1}{N}\sum_{k=1}^{N}\mathrm{E}(X_k)\right)=\eta. \tag{5.12}$$

However, the theorem is also true for (5.9). This follows from the observation that for scalars $\alpha,\eta\in[0,1]$ and for another scalar $\beta\geq 1$, we have

$$\begin{aligned} D(\alpha||\beta\,\eta) &= D(\alpha||\eta)+\alpha\log\frac{1}{\beta} \\ &\leq D(\alpha||\eta). \end{aligned} \tag{5.13}$$

Now, we will use this theorem to prove the corollary that we state below.

Corollary 22. *Let d and N be positive integers, let $\eta,\varepsilon\in[0,1]$ with $\eta>\varepsilon$ be real numbers, and let $X_1,\ldots,X_N$ be independent and identically distributed operator-valued random variables having the following properties:*

1. *Each X_k takes $d\times d$ positive semidefinite operator values satisfying $X_k\leq\mathbb{1}$.*
2. *The minimum eigenvalue of the expected operator $\mathrm{E}(X_k)$ satisfies $\lambda_{\min}(\mathrm{E}(X_k))\geq\eta$.*

It is the case that

$$\Pr\left(\lambda_{\min}\left(\frac{X_1+\cdots+X_N}{N}\right)<\eta-\varepsilon\right)\leq d\exp(-2N\varepsilon^2). \tag{5.14}$$

Proof. Observe that

$$\eta>\eta-\varepsilon>0, \tag{5.15}$$

and that, since the random variables are identically distributed,

$$\lambda_{\min}\left(\frac{1}{N}\sum_{k=1}^{N}\mathrm{E}(X_k)\right)=\lambda_{\min}(\mathrm{E}(X_k))\geq\eta, \tag{5.16}$$

for each $k\in[N]$. Now, by directly applying Theorem (20) (by taking α as $\eta-\varepsilon$), we get that

$$\Pr\left(\lambda_{\min}\left(\frac{X_1+\cdots+X_N}{N}\right)<\eta-\varepsilon\right)\leq d\exp(-N\cdot D(\eta-\varepsilon||\eta)). \tag{5.17}$$

Noting from (5.8) that

$$\exp\left(-N \cdot D(\eta - \varepsilon || \eta)\right) \leq \exp\left(-\frac{2}{\ln 2} N\varepsilon^2\right) \leq \exp(-2N\varepsilon^2), \tag{5.18}$$

where the last inequality follows from the fact that

$$\exp\left(-\frac{2}{\ln 2}\right) \approx 0.0558 < \exp(-2) \approx 0.135, \tag{5.19}$$

the proof is complete. □

5.3. Upper-bound on CQRG(1)

In this section, we will upper bound the complexity class CQRG(1). Specifically, we will show that it is contained in $\exists \cdot \mathrm{PP}$. Informally, the proof will be a generalization of the Althöfer–Lipton–Young [Alt94, LY94] technique, briefly covered in Chapter 1, of arguing for the existence of an almost-optimal distribution for the winning player which is uniform over a polynomially large set of polynomial sized strings. While the original Althöfer–Lipton–Young technique relied on Bernoulli random variables, we generalize it to operator valued random variables to account for the quantum setting. The proof of containment is divided into two initial lemmas and a final theorem where the containment is established. We will reuse the two lemmas in the next chapter, while proving that MQRG(1) is contained in $\mathrm{P} \cdot \mathrm{PP}$. The first lemma is an application of Corollary 22 to state something more relevant to our setting.

Lemma 23. *Let k and m be positive integers, let $p \in \mathcal{P}(\Sigma^k)$ be a probability distribution on k-bit strings, let S_y be a $2^m \times 2^m$ positive semidefinite operator satisfying $0 \leq S_y \leq \mathbb{1}$ for each $y \in \Sigma^k$, and let $N \geq 72(m+2)$. For strings $y_1, \ldots, y_N \in \Sigma^k$ sampled independently from the distribution p, it is the case that*

$$\Pr\left(\lambda_{min}\left(\frac{S_{y_1} + \cdots + S_{y_N}}{N}\right) < \lambda_{min}\left(\sum_{y \in \Sigma^k} p(y) S_y\right) - \frac{1}{12}\right) < \frac{1}{3}. \tag{5.20}$$

Proof. Let $X_1, \ldots, X_N$ be independent and identically distributed operator-valued random variables, each taking the value S_y with probability $p(y)$, for every $y \in \Sigma^k$. Thus, the expected value of each of these random variables is given by

$$P = \sum_{y \in \Sigma^k} p(y) S_y. \tag{5.21}$$

Observe that by taking $\eta = \lambda_{\min}(P)$ and $\varepsilon = 1/12$ and by applying Corollary 22,

$$\Pr\left(\lambda_{\min}\left(\frac{X_1 + \cdots + X_N}{N}\right) < \lambda_{\min}(P) - \frac{1}{12}\right) \leq 2^m \exp\left(-\frac{N}{72}\right) < \frac{1}{3}, \tag{5.22}$$

which proves the lemma. □

The second lemma relates minimum eigenvalues of special types of quantum measurement operators operators to PP languages. It does so with GapP functions of counting complexity. Note that the relations in (5.26), which relate the maximum eigenvalue of an operator with the trace of that operator raised to a certain power, is the main idea behind the unpublished proof of QMA $\subseteq$ PP claimed in [KW00].

Lemma 24. *Let $\{Q_x : x \in \Sigma^*\}$ be a polynomial-time generated family of quantum circuits, where each circuit Q_x takes as input a k-qubit register* Y *and an m-qubit register* B*, for polynomially bounded functions k and m, and outputs a single qubit. For each $x \in \Sigma^*$ and $y \in \Sigma^k$, define an operator*

$$S_{x,y} = \big(\langle y| \otimes \mathbb{1}_{\mathcal{B}}\big) Q_x^* \big(|1\rangle\langle 1|\big) \big(|y\rangle \otimes \mathbb{1}_{\mathcal{B}}\big). \tag{5.23}$$

For every polynomially bounded function N, there exists a language $B \in$ PP for which the following implications are true for all $x \in \Sigma^*$ and $y_1, \ldots, y_N \in \Sigma^k$:

$$\lambda_{min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \geq \frac{2}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \in B, \tag{5.24}$$

$$\lambda_{min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \leq \frac{1}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \notin B. \tag{5.25}$$

Proof. Let us give a brief informal description of the proof. First, we will relate the maximum eigenvalue of a $2^m \times 2^m$ positive semidefinite operator P with the trace of P raised to the power r, where r is a polynomially bounded function. Then, we will choose a P whose real and imaginary parts are proportional to GapP functions and use the relations to define new GapP functions whose real and imaginary parts are proportional to the real and imaginary parts of $\mathrm{Tr}(P^r)$. Finally, by choosing an appropriate r, we will relate these new GapP functions with a language B in PP such that (5.24) and (5.25) hold.

We begin by observing the following relations for any $2^m \times 2^m$ positive semidefinite operator P:

$$\lambda_{\max}(P)^r = \lambda_{\max}(P^r) \leq \mathrm{Tr}(P^r) \leq 2^m \lambda_{\max}(P^r) = 2^m \lambda_{\max}(P)^r. \tag{5.26}$$

For our case, it will be sufficient to take $r = 2m$.

Now, let us define

$$T_{x,y} = \big(\langle y| \otimes \mathbb{1}_{\mathcal{B}}\big) Q_x^* \big(|0\rangle\langle 0|\big) \big(|y\rangle \otimes \mathbb{1}_{\mathcal{B}}\big) \tag{5.27}$$

for each $x \in \Sigma^*$ and $y \in \Sigma^k$. Note that $S_{x,y}$ and $T_{x,y}$ are positive semidefinite operators which satisfy $S_{x,y} + T_{x,y} = \mathbb{1}_{\mathcal{B}}$. The implications (5.24) and (5.25) may now be written as

$$\lambda_{\max}\left(\frac{T_{x,y_1} + \cdots + T_{x,y_N}}{N}\right) \leq \frac{1}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \in B, \tag{5.28}$$

$$\lambda_{\max}\left(\frac{T_{x,y_1} + \cdots + T_{x,y_N}}{N}\right) \geq \frac{2}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \notin B. \tag{5.29}$$

Define an operator $P_{x,y_1\cdots y_N}$ as

$$P_{x,y_1\cdots y_N} = \frac{T_{x,y_1} + \cdots + T_{x,y_N}}{N}. \tag{5.30}$$

If $\lambda_{\max}(P_{x,y_1\cdots y_N}) \leq 1/3$, we have by (5.26)

$$\mathrm{Tr}(P^{2m}_{x,y_1\cdots y_N}) \leq \frac{2^m}{3^{2m}} < \frac{1}{3^m}. \tag{5.31}$$

On the other hand, if $\lambda_{\max}(P_{x,y_1\cdots y_N}) \geq 2/3$, we have by (5.26)

$$\mathrm{Tr}(P^{2m}_{x,y_1\cdots y_N}) \geq \left(\frac{2}{3}\right)^{2m} > \frac{1}{3^m}. \tag{5.32}$$

By Lemma 7 we can argue for the existence of a polynomially bounded function r and GapP functions f and g which satisfy

$$\begin{aligned} \mathrm{Re}(\langle z|T_{x,y}|w\rangle) &= \mathrm{Re}\big(\langle 0|Q_x\big(|yz\rangle\langle yw|\big)|0\rangle\big) = 2^{-r} f(x,y,z,w), \\ \mathrm{Im}(\langle z|T_{x,y}|w\rangle) &= -\,\mathrm{Im}\big(\langle 0|Q_x\big(|yz\rangle\langle yw|\big)|0\rangle\big) = 2^{-r} g(x,y,z,w), \end{aligned} \tag{5.33}$$

for all $x \in \Sigma^*$, $y \in \Sigma^k$, and $z, w \in \Sigma^m$. Define two new functions F and G such that:

$$\begin{aligned} F(x, y_1 \cdots y_N, z, w) &= f(x, y_1, z, w) + \cdots + f(x, y_N, z, w), \\ G(x, y_1 \cdots y_N, z, w) &= g(x, y_1, z, w) + \cdots + g(x, y_N, z, w), \end{aligned} \tag{5.34}$$

for all $x \in \Sigma^*$, $y_1, \ldots, y_N \in \Sigma^k$, and $z, w \in \Sigma^m$. From (5.30), we can see that F and G are GapP functions satisfying

$$\begin{aligned} F(x, y_1 \cdots y_N, z, w) &= 2^r \cdot N \cdot \mathrm{Re}(\langle z|P_{x,y_1\cdots y_N}|w\rangle), \\ G(x, y_1 \cdots y_N, z, w) &= 2^r \cdot N \cdot \mathrm{Im}(\langle z|P_{x,y_1\cdots y_N}|w\rangle). \end{aligned} \tag{5.35}$$

Applying Lemmas 4 and 6, observe that there must exist a GapP function H satisfying

$$H(x, y_1 \cdots y_N) = 2^{2rm} \cdot N^{2m} \cdot \mathrm{Tr}\big(P^{2m}_{x,y_1\cdots y_N}\big). \tag{5.36}$$

Define a new GapP function

$$K(x, y_1 \cdots y_N) = 2^{2rm} \cdot N^{2m} - 3^m \cdot H(x, y_1 \cdots y_N). \tag{5.37}$$

Observe that it is positive if $\lambda_{\max}(P_{x,y_1\cdots y_N}) \leq 1/3$, and negative if $\lambda_{\max}(P_{x,y_1\cdots y_N}) \geq 2/3$. This implies the existence of a PP language B as claimed in the theorem and the proof is complete. □

Theorem 25. $\mathrm{CQRG}(1) \subseteq \exists \cdot \mathrm{PP}$.

Proof. Let $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ be any promise problem contained in $\mathrm{CQRG}(1)$. We fix a referee for which $A \in \mathrm{CQRG}(1)_{3/4,1/4}$, and let $\{Q_x : x \in \Sigma^*\}$ be the collection of circuits describing this referee, according to Definition 10.

Let $x \in A_{\mathrm{yes}} \cup A_{\mathrm{no}}$ be any input string. First, let us consider the situation where Alice sends a deterministic string $y \in \Sigma^n$ to the referee, so that $\rho = |y\rangle\langle y|$. Let us suppose Bob plays a quantum state $\sigma \in \mathrm{D}(\mathcal{B})$. The probability that the referee gets outcomes 0 and 1 after plugging in the states of Alice and Bob is given by

$$\langle 0|Q_x(|y\rangle\langle y| \otimes \sigma)|0\rangle \quad \text{and} \quad \langle 1|Q_x(|y\rangle\langle y| \otimes \sigma)|1\rangle \tag{5.38}$$

respectively. Define an operator $S_{x,y} \in \mathrm{Pos}(\mathcal{B})$ as

$$S_{x,y} = \big(\langle y| \otimes \mathbb{1}_{\mathcal{B}}\big) Q_x^*(|1\rangle\langle 1|)\big(|y\rangle \otimes \mathbb{1}_{\mathcal{B}}\big). \tag{5.39}$$

Observe that from the properties of quantum channels elaborated in Chapter 3, especially (3.7),

$$\mathrm{Tr}\big(S_{x,y}\sigma\big) = \langle 1|Q_x(|y\rangle\langle y| \otimes \sigma)|1\rangle \tag{5.40}$$

and

$$\mathrm{Tr}\big((\mathbb{1}_{\mathcal{B}} - S_{x,y})\sigma\big) = \langle 0|Q_x(|y\rangle\langle y| \otimes \sigma)|0\rangle \tag{5.41}$$

for all $\sigma \in \mathrm{D}(\mathcal{B})$, which indicates that $S_{x,y}$ and $\mathbb{1}_{\mathcal{B}} - S_{x,y}$ are a pair of quantum measurement operators on space $\mathcal{B}$.

Note that Bob wants to minimize the probability of outcome 1 appearing. To reason about Bob's strategy, we should look at the *minimum eigenvalue* $\lambda_{\min}(S_{x,y})$ of $S_{x,y}$. This is because a large minimum eigenvalue means that Alice wins with high probability regardless of what state Bob chooses while a small minimum eigenvalue means that Bob has at least one quantum state (the density matrix corresponding to an eigenvector for the minimum eigenvalue in a spectral decomposition of $S_{x,y}$) that allows Bob to win with high probability. More technically, Bob's optimal strategy in the case that Alice plays $\rho = |y\rangle\langle y|$ is to play any state $\sigma \in \mathrm{D}(\mathcal{B})$ whose image is contained in the eigenspace of $S_{x,y}$ corresponding to the minimum eigenvalue $\lambda_{\min}(S_{x,y})$, which causes Alice to win with probability $\lambda_{\min}(S_{x,y})$ and causes Bob to win with probability $1 - \lambda_{\min}(S_{x,y})$. A way to see it is to consider a spectral decomposition for $S_{x,y}$ given by

$$S_{x,y} = \sum_{i=1}^{k} \lambda_{x,y,i} u_{x,y,i} u_{x,y,i}^*, \tag{5.42}$$

with $\{u_{x,y,i} : i \in [k]\}$ forming an orthonormal basis for $\mathcal{B}$ and $\{\lambda_{x,y,i} : i \in [k]\}$ being the eigenvalues of $S_{x,y}$, where the eigenvalues may not all be unique. Now, consider the set

$$T = \{j : j \in [k],\ \lambda_{x,y,j} = \lambda_{\min}(S_{x,y})\}. \tag{5.43}$$

Observe that a choice of $\sigma \in \mathrm{D}(\mathcal{B})$ is

$$\sigma_{x,y} = \sum_{i \in T} \mu_{x,y,i} u_{x,y,i} u_{x,y,i}^*, \tag{5.44}$$

where each $\mu_i \geq 0$ and

$$\sum_{i \in T} \mu_{x,y,i} = 1. \tag{5.45}$$

It is easy to verify that $\sigma_{x,y}$ is an operator whose image is contained in the eigenspace of $S_{x,y}$ corresponding to the minimum eigenvalue $\lambda_{\min}(S_{x,y})$ and, as described above, represents an optimal play for Bob.

In the general case, Alice will not play a single deterministic string, but will instead play a probability distribution $p \in \mathcal{P}(\Sigma^n)$ over a lot of deterministic strings. For this general strategy, the resulting measurement operator on Bob's space becomes

$$\sum_{y \in \Sigma^n} p(y) S_{x,y}. \tag{5.46}$$

Then, the probability that Alice wins when she plays a distribution $p \in \mathcal{P}(\Sigma^n)$, and Bob plays optimally against this distribution, is given by the expression

$$\lambda_{\min}\left(\sum_{y \in \Sigma^n} p(y) S_{x,y} \right). \tag{5.47}$$

Now, the task of determining whether $x \in A_{\text{yes}}$ or $x \in A_{\text{no}}$ reduces to the task of determining whether there exists a distribution $p \in \mathcal{P}(\Sigma^n)$ for which the minimum eigenvalue (5.47) is at least $3/4$ or whether the minimum eigenvalue is at most $1/4$ for all choices of $p \in \mathcal{P}(\Sigma^n)$.

We will prove that the decision problem mentioned above is contained in $\exists \cdot \mathrm{PP}$. Qualitatively, the $\exists$ operator represents the presence or absence of an optimal distribution for which the minimum eigenvalue (5.47) is large and the PP helps us to estimate that minimum eigenvalue. A problem that arises for this case is that the $\exists$ operator requires the optimal distribution to have a polynomial-length description. However, Alice's optimal distribution does not generally have a polynomial length description. Instead, in the general case, a distribution can be explicitly specified only by an exponential-length description, assuming each individual probability is specified upto polynomial precision.

To overcome this challenge, we consider a variant of the Althöfer–Lipton–Young technique. Instead of specifying a distribution $p \in \mathcal{P}(\Sigma^n)$ with exponential support, we will choose a different distribution $q \in \mathcal{P}(\Sigma^n)$ as follows. Consider the N-tuple of strings $(y_1, \ldots, y_N)$ where $N(|x|)$ is a polynomially bounded function and the strings $(y_1, \ldots, y_N)$ represent N deterministic strings that Alice can play. This N-tuple represents the distribution $q \in \mathcal{P}(\Sigma^n)$, where $q(y)$ is found by choosing an index $j \in \{1, \ldots, N\}$ uniformly at random and then outputting the string y_j. More technically, $q \in \mathcal{P}(\Sigma^n)$ represented by the N-tuple $(y_1, \ldots, y_N)$ is given by

$$q(y) = \frac{\left| \{ j \in \{1, \ldots, N\} \,:\, y = y_j \} \right|}{N} \tag{5.48}$$

for each $y \in \Sigma^n$. As one would expect, most choices of $q \in \mathcal{P}(\Sigma^n)$ are very bad approximations for the distribution p. But the existence of an optimal distribution for which the minimum eigenvalue (5.47) is large implies the existence of an N-tuple $(y_1, \ldots, y_N)$ (and hence, a distribution q) for which

$$\lambda_{\min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \tag{5.49}$$

is also sufficiently large, provided we choose a large but still polynomially bounded N. This is formally shown in Lemma 23.

Specifically, Lemma 23 implies that by choosing $N = 72(m+2)$, where m is the number of qubits of B, we find that if the minimum eigenvalue (5.47) is at least $3/4$, then with probability at least $2/3$ over the random choices of $y_1, \ldots, y_N$, the minimum eigenvalue (5.49) is at least $2/3$. By the probabilistic method, this implies the existence of at least one N-tuple $(y_1, \ldots, y_N)$ for which the minimum eigenvalue (5.49) is at least $2/3$.

On the other hand, if $x \in A_{\text{no}}$, then the minimum eigenvalue (5.47) is at most $1/4$ for all choices of $p \in \mathcal{P}(\Sigma^n)$, and therefore it holds that

$$\lambda_{\min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \leq \frac{1}{4} < \frac{1}{3} \tag{5.50}$$

for all N-tuples $(y_1, \ldots, y_N)$. This follows from the definition of CQRG(1) and the fact that the distribution q defined by (5.48) is simply one example of a distribution in $\mathcal{P}(\Sigma^n)$.

Finally, we apply Lemma 24 to argue that there exists a language $B \in \text{PP}$ such that if the minimum eigenvalue (5.49) is at least $2/3$, then $(x, y_1 \cdots y_N) \in B$, while if this minimum eigenvalue is at most $1/3$, then $(x, y_1 \cdots y_N) \notin B$. As a consequence of this fact, if $x \in A_{\text{yes}}$, then there exists a string $y_1 \cdots y_N \in \Sigma^{nN}$ such that $(x, y_1 \cdots y_N) \in B$, while if $x \in A_{\text{no}}$, then for every string $y_1 \cdots y_N \in \Sigma^{nN}$ it is the case that $(x, y_1 \cdots y_N) \notin B$. It now follows that $A \in \exists \cdot \text{PP}$. □

Chapter 6

Results - Part II

We will now prove that MQRG(1) is contained in $\mathrm{P} \cdot \mathrm{PP}$. The proof will make use of the two lemmas proved in the last chapter, and also a variant of Hoeffding's lemma that we prove in Appendix A. Additionally, the proof will involve an intermediate complexity class $\mathrm{QMA} \cdot \mathrm{PP}$, or the QMA operator acting on the class PP. Just as we did for the $\exists$ and P operators, we will prove certain general facts about the QMA-operator in the next section, before we start our actual proof.

6.1. Properties of the QMA operator

In this section, we will define the class $\mathrm{QMA} \cdot \mathcal{C}$, or the QMA operator applied to a complexity class $\mathcal{C}$. Then, we will use properties of $\mathrm{Gap} \cdot \mathcal{C}$ functions, proved in Chapter 2, to prove that $\mathrm{QMA} \cdot \mathcal{C}$ is contained in $\mathrm{P} \cdot \mathcal{C}$, for a complexity class $\mathcal{C}$ closed under joins and truth table reductions. Since PP is closed under joins and truth table reductions, it follows that $\mathrm{QMA} \cdot \mathrm{PP}$ is contained in $\mathrm{P} \cdot \mathrm{PP}$.

Definition 26. For a given complexity class $\mathcal{C}$, the complexity class $\mathrm{QMA} \cdot \mathcal{C}$ contains all promise problems $A = (A_{\mathrm{yes}}, A_{\mathrm{no}})$ for which there exists a polynomial-time generated family of quantum circuits $\{P_x : x \in \Sigma^*\}$, where each P_x takes $n = n(|x|)$ input qubits and outputs $k = k(|x|)$ qubits, along with a language $B \in \mathcal{C}$, such that the following implications hold.

1. If $x \in A_{\mathrm{yes}}$, then there exists a density operator ρ on n qubits for which
$$\Pr\big(P_x(\rho) \in B\big) \geq \frac{2}{3}. \tag{6.1}$$
2. If $x \in A_{\mathrm{no}}$, then for every density operator ρ on n qubits,
$$\Pr\big(P_x(\rho) \in B\big) \leq \frac{1}{3}. \tag{6.2}$$

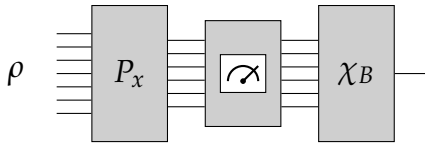

Figure 6.1: A pictorial description of QMA · $\mathcal{C}$. As per Definition 26, we are interested in the probability that the output of a circuit P_x, measured with respect to the standard basis, is contained in the language B, assuming the input is ρ.

The notation $P_x(\rho) \in B$ means that the quantum circuit P_x is applied to the state ρ, the output qubits are measured in the standard basis, and the binary string that is the measurement outcome is contained in the language B. Figure 6.1 illustrates this fact, with χ_B being the characteristic function of B on inputs of length k.

Theorem 27. *If $\mathcal{C}$ is a nontrivial complexity class of languages that is closed under joins and truth-table reductions, then* QMA · $\mathcal{C} \subseteq$ P · $\mathcal{C}$.

Proof. Let $A = (A_{\text{yes}}, A_{\text{no}}) \in$ QMA · $\mathcal{C}$ be a promise problem. Let $\{P_x : x \in \Sigma^*\}$ be a polynomial-time generated family of quantum circuits and let $B \in \mathcal{C}$ be a language that together establish the fact that $A \in$ QMA · $\mathcal{C}$ according to Definition 26.

Applying Lemma 7, we can argue for the existence of a polynomially bounded function r and GapP functions f_0 and f_1 such that

$$\begin{aligned} \text{Re}\big(\langle u|P_x(|z\rangle\langle w|)|v\rangle\big) &= 2^{-r} f_0(x,z,w,u,v), \\ \text{Im}\big(\langle u|P_x(|z\rangle\langle w|)|v\rangle\big) &= 2^{-r} f_1(x,z,w,u,v), \end{aligned} \tag{6.3}$$

for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u, v \in \Sigma^k$. Define

$$\begin{aligned} g_0(x,z,w,u) &= \begin{cases} f_0(x,z,w,u,u) & \text{if } u \in B \\ 0 & \text{if } u \notin B, \end{cases} \\ g_1(x,z,w,u) &= \begin{cases} f_1(x,z,w,u,u) & \text{if } u \in B \\ 0 & \text{if } u \notin B, \end{cases} \end{aligned} \tag{6.4}$$

for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u \in \Sigma^k$. We will now show that by the nontriviality and closure of $\mathcal{C}$ under joins and truth table reductions and the fact that P is contained in $\mathcal{C}$, g_0 and g_1 are contained in Gap · $\mathcal{C}$. We will show this explicitly for g_0 and note that the inclusion for g_1 follows a similar process.

Note that since $\mathcal{C}$ is non-trivial and closed under Karp (polynomial-time) reductions, we have that P $\subseteq \mathcal{C}$. By the assumption that $f_0 \in$ GapP, and the fact that P $\subseteq \mathcal{C}$, there exists a polynomially bounded function q and languages $A_0, A_1 \in \mathcal{C}$ such that

$$\begin{aligned} f_0(x,z,w,u,v) = &\left|\left\{y \in \Sigma^{q(|\langle x,z,w,u,v\rangle|)} : \langle x,z,w,u,v,y\rangle \in A_0\right\}\right| \\ &- \left|\left\{y \in \Sigma^{q(|\langle x,z,w,u,v\rangle|)} : \langle x,z,w,u,v,y\rangle \in A_1\right\}\right|, \end{aligned} \tag{6.5}$$

for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u \in \Sigma^k$. Define a polynomially bounded function s such that

$$s(|\langle x, z, w, u\rangle|) = q(|\langle x, z, w, u, u\rangle|). \tag{6.6}$$

Now, define languages B_0, B_1, and B_2 as follows

$$\begin{aligned} B_0 &= \big\{\langle x, z, w, u, y\rangle \,:\, y \in \Sigma^{s(|\langle x,z,w,u\rangle|)},\ \langle x, z, w, u, u, y\rangle \in A_0\big\}, \\ B_1 &= \big\{\langle x, z, w, u, y\rangle \,:\, y \in \Sigma^{s(|\langle x,z,w,u\rangle|)},\ \langle x, z, w, u, u, y\rangle \in A_1\big\}, \\ B_2 &= \big\{\langle x, z, w, u, y\rangle \,:\, y \in \Sigma^{s(|\langle x,z,w,u\rangle|)},\ u \in B\big\}. \end{aligned} \tag{6.7}$$

Since $\mathcal{C}$ is closed under Karp reductions, it immediately follows that $B_0, B_1, B_2 \in \mathcal{C}$. Since $\mathcal{C}$ is also closed under joins and truth table reductions (and hence, under intersections), it follows that $D_0 = B_0 \cap B_2$ and $D_1 = B_1 \cap B_2$ are contained $\mathcal{C}$. It is easy to see that

$$\begin{aligned} g_0(x, z, w, u) = \big|\big\{y \in \Sigma^{s(|\langle x,z,w,u\rangle|)} \,:\, \langle x, z, w, u, y\rangle \in D_0\big\}\big| \\ - \big|\big\{y \in \Sigma^{s(|\langle x,z,w,u\rangle|)} \,:\, \langle x, z, w, u, y\rangle \in D_1\big\}\big|, \end{aligned} \tag{6.8}$$

for all $x \in \Sigma^*$, $z, w \in \Sigma^n$, and $u \in \Sigma^k$. From Definition 2, it now follows that g_0 is a Gap $\cdot\, \mathcal{C}$ function. As stated before, by a very similar argument it follows that g_1 is also a Gap $\cdot\, \mathcal{C}$ function.

Now we define

$$\begin{aligned} F_0(x, z, w) &= \sum_{u \in \Sigma^k} g_0(x, z, w, u), \\ F_1(x, z, w) &= -\sum_{u \in \Sigma^k} g_1(x, z, w, u), \end{aligned} \tag{6.9}$$

for all $x \in \Sigma^*$ and $z, w \in \Sigma^n$. By Lemma 4 it follows that $F_0, F_1 \in$ Gap $\cdot\, \mathcal{C}$. Observe that from the definition of the adjoint of a map from (3.7), and from (6.3), (6.4) and (6.8),

$$\begin{aligned} \mathrm{Re}\big(\langle w|R_x|z\rangle\big) &= 2^{-r} F_0(x, z, w), \\ \mathrm{Im}\big(\langle w|R_x|z\rangle\big) &= 2^{-r} F_1(x, z, w) \end{aligned} \tag{6.10}$$

for all $x \in \Sigma^*$ and $z, w \in \Sigma^n$, where

$$R_x = \sum_{u \in \Sigma^k \cap B} P_x^*\big(|u\rangle\langle u|\big). \tag{6.11}$$

Now let us take into account the two cases $x \in A_{\mathrm{yes}}$ and $x \in A_{\mathrm{no}}$. If $x \in A_{\mathrm{yes}}$ then $\lambda_{\max}(R_x) \geq 2/3$, while if $x \in A_{\mathrm{no}}$ then $\lambda_{\max}(R_x) \leq 1/3$. Observe that as R_x is a $2^n \times 2^n$ positive semidefinite operator. Hence, similar to equation (5.26) in the proof of Lemma 24, we have that

$$\lambda_{\max}(R_x)^{n+1} = \lambda_{\max}(R_x^{n+1}) \leq \mathrm{Tr}(R_x^{n+1}) \leq 2^n \lambda_{\max}(R_x^{n+1}) = 2^n \lambda_{\max}(R_x)^{n+1}. \tag{6.12}$$

By Lemma 6 it holds that there exists a Gap $\cdot\, \mathcal{C}$ function G such that:

1. If $x \in A_{\text{yes}}$ then

$$G(x) = 2^{(n+1)r} \operatorname{Tr}(R_x^{n+1}) \geq \frac{2^{(n+1)r+n+1}}{3^{n+1}} \tag{6.13}$$

2. If $x \in A_{\text{no}}$ then

$$G(x) = 2^{(n+1)r} \operatorname{Tr}(R_x^{n+1}) \leq \frac{2^{(n+1)r+n}}{3^{n+1}}. \tag{6.14}$$

The Gap $\cdot \mathcal{C}$ function

$$H(x) = 3^{n+1} G(x) - 2^{(n+1)r+n} \tag{6.15}$$

thus satisfies $H(x) > 0$ when $x \in A_{\text{yes}}$ and $H(x) \leq 0$ when $x \in A_{\text{no}}$. By Proposition 3 it holds that $A \in \mathrm{P} \cdot \mathcal{C}$. □

6.2. Upper-bound on MQRG(1)

In this section, we prove that MQRG(1) is contained in QMA $\cdot$ PP. Combining this fact with the fact that QMA $\cdot$ PP is contained in P $\cdot$ PP, which we proved in the earlier section, will prove that MQRG(1) is contained in P $\cdot$ PP.

Theorem 28. MQRG(1) $\subseteq$ QMA $\cdot$ PP.

Proof. Consider any promise problem $A = (A_{\text{yes}}, A_{\text{no}})$ in MQRG(1). Let us and fix a referee for which $A \in \mathrm{MQRG}(1)_{3/4,1/4}$. Let $\{P_x : x \in \Sigma^*\}$ and $\{Q_x : x \in \Sigma^*\}$ be a collection of circuits that describe this referee, as per Definition 11. Just like in the proof of Theorem 25, define an operator

$$S_{x,y} = \big(\langle y| \otimes \mathbb{1}_{\mathcal{B}}\big) Q_x^*(|1\rangle\langle 1|)\big(|y\rangle \otimes \mathbb{1}_{\mathcal{B}}\big) \tag{6.16}$$

for each $x \in \Sigma^*$ and $y \in \Sigma^k$. If $x \in A_{\text{yes}}$, there exists a state $\rho \in \mathrm{D}(\mathcal{A})$ such that

$$\lambda_{\min}\left(\sum_{y \in \Sigma^k} \langle y|P_x(\rho)|y\rangle S_{x,y}\right) \geq \frac{3}{4}, \tag{6.17}$$

whereas, if $x \in A_{\text{no}}$, we have that

$$\lambda_{\min}\left(\sum_{y \in \Sigma^k} \langle y|P_x(\rho)|y\rangle S_{x,y}\right) \leq \frac{1}{4} \tag{6.18}$$

for every $\rho \in \mathrm{D}(\mathcal{A})$.

Now, let us define a polynomially bounded function $N = 72(m+2)$. By applying Lemma 24, we can argue for the existence of a language $B \in \mathrm{PP}$ such that for all $x \in \Sigma^*$ and $y_1, \ldots, y_N \in \Sigma^k$:

$$\lambda_{\min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \geq \frac{2}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \in B, \tag{6.19}$$

$$\lambda_{\min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \leq \frac{1}{3} \quad \Rightarrow \quad (x, y_1 \cdots y_N) \notin B. \tag{6.20}$$

Finally, for each input x, let us construct a circuit K_x as follows. K_x takes as input N registers $(\mathsf{A}_1, \ldots, \mathsf{A}_N)$, each consisting of n qubits, and outputs $N+1$ registers $(\mathsf{X}, \mathsf{Y}_1, \ldots, \mathsf{Y}_N)$. The register X is initialized to the state $|x\rangle\langle x|$, so that it hard-codes the input string x, and each register Y_j is obtained by independently applying the circuit P_x to A_j. More technically, one could write

$$K_x = |x\rangle\langle x| \otimes P_x^{\otimes N}, \tag{6.21}$$

where the state $|x\rangle\langle x|$ is identified with the channel that inputs nothing and outputs the state $|x\rangle\langle x|$.

Note that to prove that the promise problem A is contained in QMA $\cdot$ PP, it is sufficient to prove two things:

Completeness. If it is the case that $x \in A_{\text{yes}}$, then there must exist a state $\xi \in \mathrm{D}(\mathcal{A}^{\otimes N})$ such that

$$\Pr(K_x(\xi) \in B) \geq \frac{2}{3}. \tag{6.22}$$

Soundness. If it is the case that $x \in A_{\text{no}}$, then for every state $\xi \in \mathrm{D}(\mathcal{A}^{\otimes N})$ it must be that

$$\Pr(K_x(\xi) \in B) \leq \frac{1}{3}. \tag{6.23}$$

The proof of completeness is similar to the proof of Theorem 25. Let $\rho \in \mathrm{D}(\mathcal{A})$ be any state for which (6.17) is satisfied, and let $\xi = \rho^{\otimes N}$. Observe that the output of $K_x(\xi)$ is given by $(x, y_1 \cdots y_N)$, for $y_1, \ldots, y_N \in \Sigma^k$ sampled independently from the distribution

$$p(y) = \langle y|P_x(\rho)|y\rangle. \tag{6.24}$$

It follows by Lemma 23 that

$$\Pr(K_x(\xi) \in B) \geq \frac{2}{3}. \tag{6.25}$$

For the proof of soundness, we consider the case that the state $\xi \in \mathrm{D}(\mathcal{A}^{\otimes N})$ does not take a product form. To give an overview of the proof, we will show that if $y_1, \ldots, y_N$ are randomly selected according to the distribution that assigns the probability

$$\big\langle y_1 \cdots y_N \big| P_x^{\otimes N}(\xi) \big| y_1 \cdots y_N \big\rangle \tag{6.26}$$

to each tuple $(y_1, \ldots, y_N)$, then

$$\Pr\left(\lambda_{\min}\left(\frac{S_{x,y_1} + \cdots + S_{x,y_N}}{N}\right) \leq \frac{1}{3}\right) \geq \frac{2}{3}. \tag{6.27}$$

This immediately implies that $\Pr(K_x(\xi) \in B) \leq 1/3$ by (6.20). Now, to show this, choose a density operator $\sigma \in \mathrm{D}(\mathcal{B})$ for which

$$\sum_{y \in \Sigma^k} \langle y|P_x(\rho)|y\rangle \operatorname{Tr}\big(S_{x,y}\sigma\big) \leq \frac{1}{4} \tag{6.28}$$

for all $\rho \in \mathrm{D}(\mathcal{A})$. The existence of such a σ is made possible by Sion's min-max theorem under the assumption (6.18), and the observation, by the linearity of trace, that

$$\mathrm{Tr}\left(\sum_{y\in\Sigma^k} \langle y|P_x(\rho)|y\rangle S_{x,y}\sigma\right) = \sum_{y\in\Sigma^k} \langle y|P_x(\rho)|y\rangle \,\mathrm{Tr}\big(S_{x,y}\sigma\big). \tag{6.29}$$

In essence, the assumption (6.18) says that if the minimum eigenvalue of the operator

$$\sum_{y\in\Sigma^k} \langle y|P_x(\rho)|y\rangle S_{x,y} \tag{6.30}$$

is less than $1/4$, then there exists a state σ for which the quantity

$$\mathrm{Tr}\left(\sum_{y\in\Sigma^k} \langle y|P_x(\rho)|y\rangle S_{x,y}\sigma\right), \tag{6.31}$$

which is essentially the inner product of (6.30) with σ, is less than $1/4$. Define random variables $Z_1, \ldots, Z_N$ as

$$Z_j = \mathrm{Tr}\big(S_{x,y_j}\sigma\big) \tag{6.32}$$

for every $j \in \{1, \ldots, N\}$, assuming that $y_1, \ldots, y_N$ are chosen at random as above. It suffices to prove that

$$\Pr\left(\frac{Z_1 + \cdots + Z_N}{N} \leq \frac{1}{3}\right) \geq \frac{2}{3}, \tag{6.33}$$

as we have $\lambda_{\min}(H) \leq \mathrm{Tr}(H\sigma)$ for all Hermitian operators H.

The challenge now is that the random variables $Z_1, \ldots, Z_N$ are not necessarily independent (because ξ does not necessarily have product form). Hence, we cannot use the standard form of Hoeffding's inequality to prove (6.33). However, note that $Z_1, \ldots, Z_N$ are discrete random variables that take values in the interval $[0, 1]$ and satisfy the inequality

$$\mathrm{E}(Z_j | Z_1 = \alpha_1, \ldots, Z_{j-1} = \alpha_{j-1}) \leq \frac{1}{4} \tag{6.34}$$

for all $j \in \{2, \ldots, N\}$ and $\alpha_1, \ldots, \alpha_{j-1} \in [0, 1]$ for which $\Pr(Z_1 = \alpha_1, \ldots, Z_{j-1} = \alpha_{j-1})$ is nonzero. This fact follows from the inequality (6.28), which must hold when ρ is equal to the reduced state of register A_j, conditioned on any choice of $y_1, \ldots, y_{j-1}$ (and therefore on any choice of values $Z_1 = \alpha_1, \ldots, Z_{j-1} = \alpha_{j-1}$) that appear with nonzero probability. In essence, (6.33) is a very strong condition. It implies that no matter what state Alice plays, irrespective of the state being entangled across all the N registers and the outcome of the measurement of the reduced state across one register dependent on the outcome of previous measurements of the reduced state across other registers, Bob still has a $\sigma \in \mathrm{D}(\mathcal{B})$ that allows him to win with high probability.

As explained in Appendix A, we can leverage the standard proof of Hoeffding's inequality to get a variant of Hoeffding's inequality for dependent random variables with bounded conditional expectation which establishes that

$$\Pr\left(\frac{Z_1+\cdots+Z_N}{N} \geq \frac{1}{3}\right) = \Pr\left(\frac{Z_1+\cdots+Z_N}{N} \geq \frac{1}{4}+\frac{1}{12}\right) \leq \exp\left(-\frac{2N}{144}\right) < \frac{1}{3}. \quad (6.35)$$

After obtaining this bound, the proof is complete. □

Corollary 29. $\mathrm{MQRG}(1) \subseteq \mathrm{P} \cdot \mathrm{PP}$.

Chapter 7

Conclusion

We defined two restricted versions of QRG(1), which we named CQRG(1) and MQRG(1). We proved that CQRG(1) is contained in $\exists \cdot \mathrm{PP}$ and MQRG(1) is contained in $\mathrm{P} \cdot \mathrm{PP}$.

An open problem can be to prove a better containment than PSPACE for the unrestricted version of QRG(1). Our containments put CQRG(1) and MQRG(1) in the counting hierarchy. We wonder whether QRG(1) is also in the counting hierarchy. There can be other restrictions on QRG(1), like restricting both provers to be classical. Since RG(1) is contained in $\mathrm{ZPP}^{\mathrm{NP}}$ we wonder whether QRG(1) with both provers being classical is contained in $\mathrm{ZPP}^{\mathrm{QCMA}}$. Other open problems can be to try and find oracle separations between QRG(1) and counting classes like PP, AWPP, or WAPP.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, USA, 1st edition, 2009.

[ADH97] L. Adleman, J. DeMarrais, and M. Huang. Quantum computability. *SIAM Journal on Computing*, 26(5):1524–1540, 1997.

[AKKT19] Scott Aaronson, Robin Kothari, William Kretschmer, and Justin Thaler. Quantum lower bounds for approximate counting via laurent polynomials, 2019.

[AKN98] D. Aharonov, A. Kitaev, and N. Nisan. Quantum circuits with mixed states. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 20–30, 1998.

[Alt94] I. Althöfer. On sparse approximations to randomized strategies and convex combinations. *Linear Algebra and Its Applications*, 199:339–355, 1994.

[Bab85] L. Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 421–429, 1985.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, October 1997.

[BCF+95] L. Babai, G. Cooperman, L. Finkelstein, E. Luks, and Á. Seress. Fast Monte Carlo algorithms for permutation groups. *Journal of Computer and System Sciences*, 50:296–307, 1995.

[BCWdW01] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001.

[BGM06] Elmar Böhler, Christian Glaßer, and Daniel Meister. Error-bounded probabilistic computations between MA and AM. *Journal of Computer and System Sciences*, 72(6):1043–1076, September 2006.

[BM88] L. Babai and S. Moran. Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988.

[Bor77] A. Borodin. On relating time and space to size and depth. *SIAM Journal on Computing*, 6:733–744, 1977.

[BRS95] R. Beigel, N. Reingold, and D. Spielman. PP is closed under intersection. *Journal of Computer and System Sciences*, 50(2):191–202, 1995.

[BW16] R. Bhattacharya and E. Waymire. *A Basic Course in Probability Theory*. Springer, second edition, 2016.

[Cai07] J.-Y. Cai. $S_2^p \subseteq ZPP^{NP}$. *Journal of Computer and System Sciences*, 73(1):25–35, 2007.

[Can96] R. Canetti. More on BPP and the polynomial-time hierarchy. *Information Processing Letters*, 57(5):237–241, 1996.

[CFLS95] A. Condon, J. Feigenbaum, C. Lund, and P. Shor. Probabilistically checkable debate systems and approximation algorithms for PSPACE-hard functions. *Chicago Journal of Theoretical Computer Science*, 1995:4, 1995.

[CFLS97] A. Condon, J. Feigenbaum, C. Lund, and P. Shor. Random debaters and the hardness of approximating stochastic functions. *SIAM Journal on Computing*, 26(2):369–400, 1997.

[Cho75] M.-D. Choi. Completely positive linear maps on complex matrices. *Linear Algebra and Its Applications*, 10(3):285–290, 1975.

[CHTW04] R. Cleve, P. Høyer, B. Toner, and J. Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of the 19th Annual IEEE Conference on Computational Complexity*, pages 236–249, 2004.

[CKS81] A. Chandra, D. Kozen, and L. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[Con87] A. Condon. *Computational Models of Games*. PhD thesis, University of Washington, 1987.

[DH09] E. Demaine and R. Hearn. Playing games with algorithms: Algorithmic combinatorial game theory. In M. Albert and R. Nowakowski, editors, *Games of No Chance 3*, pages 3–56. Cambridge University Press, 2009.

[DHM+05] Christopher M. Dawson, Andrew P. Hines, Duncan Mortimer, Henry L. Haselgrove, Michael A. Nielsen, and Tobias J. Osborne. Quantum computing and polynomial equations over the finite field $\mathbb{Z}_2$. *Quantum Info. Comput.*, 5(2):102–112, March 2005.

[ESY84] Shimon Even, Alan L. Selman, and Yacov Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, May 1984.

[Fan53] K. Fan. Minimax theorems. *Proceedings of the National Academy of Sciences*, 39:42–47, 1953.

[FB02] Gary William Flake and Eric B. Baum. Rush hour is PSPACE-complete, or "why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1-2):895–911, January 2002.

[FFK94] S. Fenner, L. Fortnow, and S. Kurtz. Gap-definable counting classes. *Journal of Computer and System Sciences*, 48:116–148, 1994.

[FIKU08] L. Fortnow, R. Impagliazzo, V. Kabanets, and C. Umans. On the complexity of succinct zero-sum games. *Computational Complexity*, 17:353–376, 2008.

[FK97] U. Feige and J. Kilian. Making games short. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 506–516, 1997.

[FKS95] J. Feigenbaum, D. Koller, and P. Shor. A game-theoretic classification of interactive complexity classes. In *Proceedings of the 10th Conference on Structure in Complexity Theory*, pages 227–237, 1995.

[FL81] Aviezri S. Fraenkel and David Lichtenstein. Computing a perfect strategy for nn chess requires time exponential in n. In *Automata, Languages and Programming*, pages 278–293. Springer Berlin Heidelberg, 1981.

[For97] L. Fortnow. Counting complexity. In L. Hemaspaandra and A. Selman, editors, *Complexity Theory Retrospective II*, pages 81–107. Springer, 1997.

[FR96] L. Fortnow and N. Reingold. PP is closed under trhuth-table reductions. *Information and Computation*, 124(1):1–6, 1996.

[FR99] L. Fortnow and J. Rogers. Complexity limitations on quantum computation. *Journal of Computer and System Sciences*, 59(2):240–252, 1999.

[FS92] U. Feige and A. Shamir. Multi-oracle interactive protocols with constant space verifiers. *Journal of Computer and System Sciences*, 44:259–271, 1992.

[FST90] U. Feige, A. Shamir, and M. Tennenholtz. The noisy oracle problem. In *Advances in Cryptology – Proceedings of Crypto'88*, volume 403 of *Lecture Notes in Computer Science*, pages 284–296. Springer–Verlag, 1990.

[FvdG99] C. Fuchs and J. van de Graaf. Cryptographic distinguishability measures for quantum-mechanical states. *IEEE Transactions on Information Theory*, 45(4):1216–1227, 1999.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pages 291–304, 1985.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[Gol06] Oded Goldreich. On promise problems: A survey. In *Lecture Notes in Computer Science*, pages 254–290. Springer Berlin Heidelberg, 2006.

[Gol08] Oded Goldreich. Computational complexity: A conceptual perspective. *SIGACT News*, 39(3):35–39, September 2008.

[GS89] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 73–90. JAI Press, 1989.

[Gut05] G. Gutoski. Upper bounds for quantum interactive proofs with competing provers. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, pages 334–343, 2005.

[GW05] G. Gutoski and J. Watrous. Quantum interactive proofs with competing provers. In *Proceedings of the 22nd Symposium on Theoretical Aspects of Computer Science*, volume 3404 of *Lecture Notes in Computer Science*, pages 605–616. Springer, 2005.

[GW07] G. Gutoski and J. Watrous. Toward a general theory of quantum games. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 565–574, 2007.

[GW13] G. Gutoski and X. Wu. Parallel approximation of min-max problems. *Computational Complexity*, 2(22):385–428, 2013.

[GW20] Soumik Ghosh and John Watrous. Complexity limitations on one-turn quantum refereed games, 2020.

[JW09] R. Jain and J. Watrous. Parallel approximation of non-interactive zero-sum quantum games. In *Proceedings of the 24th IEEE Conference on Computational Complexity*, pages 243–253, 2009.

[Kay00] Richard Kaye. Minesweeper is NP-complete. *The Mathematical Intelligencer*, 22(2):9–15, March 2000.

[KM92] D. Koller and N. Megiddo. The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior*, 4:528–552, 1992.

[KSV02] A. Kitaev, A. Shen, and M. Vyalyi. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, 2002.

[KW00] A. Kitaev and J. Watrous. Parallelization, amplification, and exponential time simulation of quantum interactive proof systems. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, pages 608–617, 2000.

[LY94] R. Lipton and N. Young. Simple strategies for large zero-sum games with applications to complexity theory. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, pages 734–740, 1994.

[MW05] C. Marriott and J. Watrous. Quantum Arthur-Merlin games. *Computational Complexity*, 14(2):122–152, 2005.

[MW18] Sanketh Menda and John Watrous. Oracle separations for quantum statistical zero-knowledge, 2018.

[NC00] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[RS98] A. Russell and R. Sundaram. Symmetric alternation captures BPP. *Computational Complexity*, 7(2):152–162, 1998.

[SC79] L. Stockmeyer and A. Chandra. Provably difficult combinatorial games. *SIAM Journal on Computing*, 8(2):151–174, 1979.

[Tro12] J. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

[Wat09] J. Watrous. Quantum computational complexity. In *Encyclopedia of Complexity and System Science*. Springer, 2009.

[Wat18] J. Watrous. *Theory of Quantum Information*. Cambridge University Press, 2018.

[Wil17] M. Wilde. *Quantum Information Theory*. Cambridge University Press, second edition, 2017.

Appendices

Appendix A

A modified Hoeffding's inequality

In our proof of Theorem 28 we used a slight modified version of Hoeffding's inequality. In our version, the random variables were no longer independent. However, we did have an upper bound on the conditional expectation value of each of those random variables. It is likely that such a modification has been used before in literature, but we have not found a suitable reference. A similar bound is proved in [BCF+95] (see Corollary 2.2 of this paper) for Bernoulli random variables, with a given lower bound on the conditional expectation of each of those variables. For our case, we need something that holds more generally for discrete random variables, with a given upper bound on the conditional expectation of each of those variables.

Note that we can slightly modify the standard proof of the usual Hoeffding's inequality to get our bound. We will start with Hoeffding's lemma: it is a common lemma and a proof of it can be found in [BW16], and other references. Hence, we omit the proof here.

Lemma 30 (Hoeffding's lemma)**.** *Let X be a random variable which takes values in $[\alpha, \beta]$, for real numbers $\alpha < \beta$, and assume $\mathrm{E}(X) \leq 0$. For every $\lambda > 0$, we have*

$$\mathrm{E}(\exp(\lambda X)) \leq \exp\left(\frac{\lambda^2}{8(\beta-\alpha)^2}\right). \tag{A.1}$$

Remark 31. *Usually, in the statement of this lemma, we assume that $\mathrm{E}(X) = 0$. However, the lemma is also true when $\mathrm{E}(X) \leq 0$. This follows from the observation that if $\mathrm{E}(X) \leq 0$, we have*

$$\mathrm{E}(\exp(\lambda X)) \leq \mathrm{E}(\exp(\lambda(X - \mathrm{E}(X)))). \tag{A.2}$$

The next lemma we prove will be used in the final proof of Hoeffding's inequality. We will prove the lemma for discrete random variables, which is enough for our needs.

Lemma 32. *Let X and Y be discrete random variables taking values in $[\alpha, \beta]$ for real numbers $\alpha < \beta$, and assume that $\mathrm{E}(Y \mid X) \leq 0$. For every $\lambda > 0$ it is the case that*

$$\mathrm{E}(\exp(\lambda(X+Y)) \leq \exp\left(\frac{\lambda^2}{8(\beta-\alpha)^2}\right) \mathrm{E}(\exp(\lambda X)). \tag{A.3}$$

Proof. Let us write

$$\mathrm{E}(\exp(\lambda(X+Y)) = \sum_x \exp(\lambda x)\,\mathrm{E}(\exp(\lambda Y) \mid X = x)\,\Pr(X = x), \tag{A.4}$$

where we take the sum over all possible values of X. We have assumed that $\mathrm{E}(Y \mid X) \leq 0$. Hence, upon applying Hoeffding's lemma, we have

$$\begin{aligned} &\sum_x \exp(\lambda x)\,\mathrm{E}(\exp(\lambda Y) \mid X = x)\,\Pr(X = x) \\ &\quad \leq \exp\left(\frac{\lambda^2}{8(\beta-\alpha)^2}\right) \sum_x \exp(\lambda x)\,\Pr(X = x) = \exp\left(\frac{\lambda^2}{8(\beta-\alpha)^2}\right) \mathrm{E}(\exp(\lambda X)). \end{aligned} \tag{A.5}$$

This proves the lemma. □

We are now ready to prove the modified version of Hoeffding's inequality that we have used in our thesis.

Theorem 33. *Let $X_1, \ldots, X_n$ be discrete random variables taking values in $[0,1]$, let $\gamma \in [0,1]$, and assume that*

$$\mathrm{E}(X_k \mid X_1, \ldots, X_{k-1}) \leq \gamma \tag{A.6}$$

for all $k \in \{1, \ldots, n\}$. For all $\varepsilon > 0$ it is the case that

$$\Pr\big(X_1 + \cdots + X_n \geq (\gamma + \varepsilon)n\big) \leq \exp(-2n\varepsilon^2). \tag{A.7}$$

Proof. Observe that for every $\lambda > 0$, it is true that

$$\begin{aligned} &\Pr\big(X_1 + \cdots + X_n \geq (\gamma+\varepsilon)n\big) \\ &\qquad = \Pr\big(\exp\big(\lambda(X_1 + \cdots + X_n - \gamma n)\big) \geq \exp(\lambda\varepsilon n)\big) \\ &\qquad \leq \frac{\mathrm{E}\big(\exp\big(\lambda(X_1 + \cdots + X_n - \gamma n)\big)\big)}{\exp(\lambda\varepsilon n)}, \end{aligned} \tag{A.8}$$

where the last line follows from Markov's inequality. If we apply Lemma 32 iteratively, we have that

$$\mathrm{E}\big(\exp\big(\lambda(X_1 + \cdots + X_n - \gamma n)\big)\big) \leq \exp\left(\frac{n\lambda^2}{8}\right). \tag{A.9}$$

The proof now follows from choosing $\lambda = 4\varepsilon$. □