A Quantum Polynomial Hierarchy and a Simple Proof of Vyalyi's Theorem

Lieuwe Vinkhuijzen

Contents

Introduction

A major question in computational complexity theory is to understand the relation between classical computation and quantum computation: if a quantum computer can solve a problem efficiently, is there also a (potentially randomized) classical algorithm for it (in complexity-theoretic jargon, we ask: is $\mathbf{BPP} = \mathbf{BQP}$)? And if a candidate solution to a problem can be quickly verified classically, can it be quickly found by a quantum computer (is $\mathbf{NP} \subseteq \mathbf{BQP}$)?

While these questions have remained wide open, exciting progress has been made in the last two decades. For example, there is a quantum analogue of the class **NP**, called **QMA**, of problems that seem intractable to solve, but where candidate solutions can be efficiently checked on a quantum computer. Even the concept of **NP**-completeness is mirrored in a growing number of **QMA**-complete problems that arise naturally in the context of quantum mechanics. In this thesis, we explore one of these **QMA**-Complete problems, the Local Hamiltonian problem, which is a natural generalization of classical constraint satisfaction problems, in which constraints are allowed to destructively interfere with one another.

A central object of study in classical computational complexity is the polynomial hierarchy, which generalizes the concept of non-determinism. Inspired by this approach, we define a new quantum analogue of this object, which we call the quantum polynomial hierarchy, and use a novel method of relativizing the aforementioned Local Hamiltonian problem to show that each level of this hierarchy has complete problems. We use these ideas to give a shorter and simpler proof of a theorem by Vyalyi.

The similarity between the two hierarchies extends further than complete problems: we survey a host of theorems related to the classical hierarchy, and show that they mostly fall into two categories. Either a theorem has an elegant quantum analogue and its classical proof requires minimal adaptation, or adapting the proof fails for one of a variety of interesting reasons. For example, several conditional collapses of the hierarchy carry over to the quantum case, but we fail to adapt Toda's theorem because it relies on Boolean formulae having a discrete number

of solutions, whereas the set of states accepted by a quantum circuit may be uncountably infinite. This way we obtain new insights into which proofs, and which parts of those proofs, are contingent on the fact that the underlying computation is classical and not quantum, which may be valuable by themselves.

Chapter 1

Introduction to computational complexity

Computational complexity theory is the branch of computer science which will eventually resolve the **P** vs **NP** conjecture. It takes a top-down approach to computer science: rather than taking a specific problem and analyzing several algorithms which make different tradeoffs on time and memory consumption, we fix bounds on the use of resources and then associate with those bounds the infinite set of problems that can be solved by algorithms whose resource consumption obeys those bounds. The resources may be traditional - the amount of time and memory used, the number of random coins flipped - but often they are novel and esoteric-looking resources that are specific to complexity theory, like non-determinstim, alternation and interaction. This results in a vast zoo of *"complexity classes"*, corresponding to the various ways of choosing bounds on the resources. While sometimes we do classify a particular problem as belonging to a particular class, to us that will be a means to an end, namely to understand the relationship between complexity classes, with the ultimate goal of understanding whether the relationship between **P** and **NP** is equality, or inequality.

Acknowledging that complexity uses several tools less known to other areas of computer science, this chapter lays out the ones that feature in this thesis.

1.1 Complexity classes

Most of computer science is concerned with finding algorithms to solve problems; to map an input to an output. In this thesis we analyze the simplest kind of those

problems, namely the ones whose output is only a single bit: the algorithm gets some input string and its job is to output either 0 or 1, where the answer depends only on the current input and not on previous input or sensory data from the environment. We call the set of inputs for which the answer is 1 the *language* that the algorithm must accept. For example, the language PRIMES is the set of (binary strings which represent) prime numbers, and the language SAT is the set of satisfiable Boolean formulae.

Recently, Agrawal et. al [8] showed that PRIMES $\in \mathbf{P}$ by discovering a deterministic polynomial-time algorithm which, when it is given a number p, outputs 1 if p is a prime, and 0 otherwise. When it outputs 1, we say that it *accepts* the input. The set $\mathbf{P}$ consists precisely of the languages with this property. The definition of $\mathbf{P}$ is that it is the set of all languages which can be solved by a deterministic Turing Machine which halts on every input and for which there is a polynomial function $p(n)$ such that for all inputs x, the algorithm takes at most $p(\text{length}(x))$ steps. The class $\mathbf{NP}$ consists of those languages L for which there exists a polynomial-time Turing Machine M such that, if $x \in L$, then there is a y such that $M(x, y)$ accepts, whereas if $x \notin L$, then for all y, $M(x, y)$ rejects. Definitions of all complexity classes used in this thesis can be found in Section 1.3.

In spite of these robust definitions and much research, we do not know which languages are in $\mathbf{P}$ and which are not; for example, we do not know whether SAT is in $\mathbf{P}$.

1.1.1 Reductions

A typical way to measure the complexity of a problem is to find the "best" algorithm for it according to some metric, for example the fastest one. But in computational complexity theory, that metric may be one of many incomparable types of resource consumption, and time consumption is not always the most meaningful one.[1] To compare the relative difficulty of two problems, our metric of choice is reducibility. A general and natural way to establish that, for example, the problem SAT is at least as hard as PRIMES, if not harder, is to establish a reduction from PRIMES to SAT.

The simplest type of reduction is the Turing reduction. A language L *Turing-reduces* to a language K, written $L \leq_T K$ if L is decided by a Turing Machine which is allowed to make queries of the form, "*is* $s \in K$*?*" and receive the answer instantly. We say that the machine *has oracle access to* K. If the machine runs

[1] Because time consumption is not a measure which preserves the subset relation between complexity classes.

in polynomial time, we write $L \leq_T^p K$ (in the notation, the T stands for Turing and the p for polynomial). There are no restrictions on the number of queries the Turing Machine makes (except of course that this number will be bounded from above by the machine's running time). In this work we will always take K to be a computable language, but in general this is not required.

For example, the problem CLIQUE, of deciding whether a graph G has a clique of size k, can be Turing-reduced to the problem MAXCLIQUE, of deciding whether the largest clique in G has size k: one repeatedly asks, *is the largest clique of size* 1? *Is it of size* 2? And so on, up to n, at which point we will know the size m of the largest clique. If $m \geq k$, then we output *yes*; otherwise we output *no*.

A finer notion of reducibility is the Karp-reduction.[2] A language L *Karp-reduces to* K, written $L \leq_m K$, if there is a function f such that for all strings $x \in \{0,1\}^*$, $x \in L \iff f(x) \in K$. Instances of L, then, can be "rewritten" as instances of K. Put another way, a Karp reduction is a Turing reduction in which the machine is allowed to make one query to K and must copy the query's answer as its output. If the function is computed in a polynomial amount of time, we write $L \leq_m^p K$. The fact that every language in **NP** can be reduced this way to SAT is the classic Cook-Levin theorem and establishes that SAT is **NP**-complete.

1.1.2 Closure

The most elementary questions one can ask about a complexity class are what its closure properties are. For example, if L and K are in **NP**, is the language $L \cap K$ also in **NP**? In this case, the answer is *yes*, and we say that **NP** is *closed under intersection*. Similarly, a class can be closed under union.

A complexity class $\mathfrak{C}$ is closed under complement if $L \in \mathfrak{C} \iff \overline{L} \in \mathfrak{C}$. It is not known or expected, for example, that **NP** is closed under complement: for if a Boolean formula is *not* satisfiable, how could somebody convince you of that, other than by going through all possible truth assignments? We write $\textbf{co-}\mathfrak{C} = \{\overline{L} \mid L \in \mathfrak{C}\}$ to denote the set of complements of languages in $\mathfrak{C}$, for example **co-NP** contains the language of Boolean formulas that are not satisfiable and the set of graphs that do not have a Hamilton path, and so forth. It is immediate that for any complexity class, $\mathfrak{C} \cap \textbf{co-}\mathfrak{C}$ is closed under complement. We also have $\mathfrak{C} \subseteq \textbf{co-}\mathfrak{C} \iff \mathfrak{C} = \textbf{co-}\mathfrak{C} \iff \mathfrak{C} = \mathfrak{C} \cap \textbf{co-}\mathfrak{C}$.

One salient property of any complexity class $\mathfrak{C}$ is that if $K \in \mathfrak{C}$, and L is easier

[2]Some texts use the term many-one reduction, and say that PRIMES *many-one reduces* to SAT, hence the m in $\leq_m^p$.

than K, then $\mathfrak{C}$ had better contain L. Otherwise, the idea that a complexity class is a description of resource bounds and the algorithms which operate within them, is violated. This intuition is formalized when a class $\mathfrak{C}$ is closed under reductions, meaning that for every language K in $\mathfrak{C}$, $\mathfrak{C}$ also contains every language that reduces to L.

In saying this, one must be careful to specify what type of reduction one has in mind. For example, **NP** is known to be closed under Karp reductions, is expected but not known to be closed under randomized Karp reductions, but is not known or expected to be closed under Turing reductions. As another example, it is easy to see that $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is closed under Karp reductions, but it takes some work to show that $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is closed under Turing reductions, and to our knowledge this work is the first which notices that the quantum version of this theorem is true, namely $\mathbf{QMA} \cap \mathbf{co\text{-}QMA}$ is also closed under Turing reductions (though we do not wish to alarm the reader, and assure her that the proof is very readable and in fact only uses elementary linear algebra).

1.2 Oracle machines

We mentioned Turing reductions earlier, in which a machine has instant access to some (presumably difficult) language. This is an appropriate technique in the context of reductions, but it is also a useful technique in slightly deeper complexity theory. When in the 70's, researchers were trying to prove $\mathbf{P} \neq \mathbf{NP}$, but the problem wouldn't budge, a natural step was to just give the deterministic machine the SAT function for free, defining a new class called $\Delta_2^{\mathbf{P}}$, which was presumably much more powerful than **P** because we have $\mathbf{P} \subseteq \mathbf{NP} \subseteq \Delta_2^{\mathbf{P}}$. Hence it was hoped that proving $\mathbf{P} \neq \Delta_2^{\mathbf{P}}$ would be easier than proving $\mathbf{P} \neq \mathbf{NP}$.

The technique of giving algorithms instant access to a problem is called *relativization* and the language to which they gain access is called the *oracle language* or simply *the oracle*. Formally, we have to perform minor surgery on the Turing Machine model to account for the presence of oracles. All oracle Turing Machines have three additional special states $q_{query}, q_{yes}, q_{no}$ (not to be confused with q_{accept} and q_{reject}) and one additional write-only tape, called the oracle tape. When the machine enters the state q_{query}, the string s on the oracle tape is considered. If $s \in L$, where L is the oracle, the machine transitions to the state q_{yes}; otherwise, if $s \notin L$, it goes to q_{no}. Lastly, the content of the tape is erased and the head of the oracle tape is brought back to the first square. This whole procedure is counted as one time step (alternatively, we can say that deciding whether $s \in L$ is not counted towards execution time). If M is an oracle Turing Machine and L is a

language, we denote with M^L the machine having oracle access to L, e.g. M^{SAT}.

To relativize a complexity class, rather than a single Turing Machine, with a particular language L, one takes the set of all appropriate Turing Machines, gives them oracle access to L, and then considers the set of resulting languages. For example, to relativize $\mathbf{P}$ with SAT, one takes all polynomial-time deterministic oracle Turing Machines, and gives them oracle access to SAT. The resulting class is denoted $\mathbf{P}^{\mathrm{SAT}}$. The "real" class $\mathbf{P}$, in this context, is $\mathbf{P}^{\emptyset}$, i.e. we give the oracle machines access to the empty set. We sometimes speak of relativizing a class, say $\mathbf{P}$, with another class, say $\mathbf{NP}$. In this case we simply take the union of $\mathbf{P}^L$ over $L \in \mathbf{NP}$. Specifically, let $TM(\mathbf{P})$ be the set of deterministic, polynomial-time oracle Turing Machines. Then

$$\mathbf{P}^{\mathbf{NP}} \equiv \bigcup_{L \in \mathbf{NP}} \bigcup_{M \in TM(\mathbf{P})} \text{Language}\left(M^L\right) \tag{1.1}$$

There is a subtle abuse of language going on in the left side of this equation. When we put $\mathbf{NP}$ in the superscript, we mean the set of *languages* accepted by polynomial-time non-deterministic Turing Machines, but by $\mathbf{P}$ in the base, we refer to a set of oracle *Turing Machines*. The whole expression $\mathbf{P}^{\mathbf{NP}}$, therefore, means: the union, over $L \in \mathbf{NP}$, of the sets of languages accepted by deterministic, polynomial-time oracle Turing Machines with access to L. For example, the statement $\mathbf{P}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} = \mathbf{NP} \cap \mathbf{co\text{-}NP}$ unambiguously means that the class $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is closed under Turing reductions.

Sometimes we will stack complexity classes, for example in one proof we need the class $\mathbf{NP}^{\mathbf{NP}^{\mathbf{NP}}}$. A stack like $\mathfrak{A}^{\mathfrak{B}^{\mathfrak{C}}}$ is to be read from top to bottom. First we consider the set of languages accepted by $\mathfrak{C}$-machines, and give those languages to $\mathfrak{B}$-machines as oracles, resulting in the set of languages $\mathfrak{D} = \mathfrak{B}^{\mathfrak{C}}$. Finally we give all $\mathfrak{A}$-machines the languages in $\mathfrak{D}$ as oracles.

The most interesting property of relativized complexity classes is no doubt that their relationship with one another provably depends upon the oracle. Specifically, there are languages Y and N such that, provably, $\mathbf{P}^Y = \mathbf{NP}^Y$ and yet $\mathbf{P}^N \neq \mathbf{NP}^N$ [16] (Of course we do not know whether $\mathbf{P}^{\emptyset} = \mathbf{NP}^{\emptyset}$!). Remember, therefore, that complexity classes are defined *by reference* to the underlying Turing Machine model, and are not defined *by value* as a set of languages.

Most theorems in complexity theory *relativize*, meaning that their theorem statement is true relative to every oracle language. For example, the time hierarchy theorem, proved in the seminal paper [18] by Hartmanis and Stearns which first established computational complexity, states (roughly) that Turing Machines given

$O(n^3)$ time solve a strictly larger set of languages than Turing Machines given only $O(n^2)$ time. The proof is simply a time-bounded version of the proof of the undecidability of the Halting Problem, and uses as its main ingredient the fact that there is a universal Turing Machine. But this is true in every oracle world: given oracle access to a language L, there is a universal Turing Machine capable of simulating all other machines that have oracle access to L, so this ingredient relativizes.

In some sense, these theorems are widely applicable: they are always true, even in the oracle model, relative to an uncountable number of languages! On the other hand, there is a certain simplicity to them: They fail to observe any deep or fundamental fact about the nature of computation that holds in the standard Turing Machine model, but not in the oracle model, and hence are not sensitive to the presence or absence of oracles. The fact that the oracles Y and N above exist means that, whatever the proof of $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$, it cannot be one of these "simple" theorems, and must use some nontrivial fact about computation other than the existence of a universal Turing Machine. Hence relativization is known as one of the three "formal barriers" towards $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$, along with the barriers of Natural Proofs [50] and Algebrization [26]. See [17] and [25] for discussion and background.

Some relativizing theorems involve stacks of complexity classes. For example, Toda's theorem states that $\mathbf{NP}^{\mathbf{NP}} \subseteq \mathbf{P}^{\mathbf{PP}}$. To say that Toda's Theorem relativizes is to say:

$$\text{For all languages } L \subseteq \{0,1\}^\star\text{: } \mathbf{NP}^{\mathbf{NP}^L} \subseteq \mathbf{P}^{\mathbf{PP}^L} \tag{1.2}$$

For all complexity classes $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$, we have:

$$\text{If } \mathfrak{A} \subseteq \mathfrak{B} \text{ then } \mathfrak{C}^{\mathfrak{A}} \subseteq \mathfrak{C}^{\mathfrak{B}} \tag{1.3}$$

$$\text{If } \mathfrak{A} = \mathfrak{B} \text{ then } \mathfrak{C}^{\mathfrak{A}} = \mathfrak{C}^{\mathfrak{B}} \tag{1.4}$$

Note that the same does *not* hold for languages. If $L \subseteq K$ then it does not follow that $\mathfrak{C}^L \subseteq \mathfrak{C}^K$ (for example, take $K = \{0,1\}^*$ and L some undecidable language). However, if we consider not the $\subseteq$ subset relation but the $\leq$ reduction relation, then the obvious implications do hold. Namely, if $L \leq_{\mathfrak{R}} K$ then $\mathfrak{C}^L \subseteq \mathfrak{C}^K$ where $\mathfrak{R}$ is some suitable notion of reduction (e.g. polynomial-time Karp reductions) and $\mathfrak{C}$ is closed under $\mathfrak{R}$-reductions. It is not true in general that if $\mathfrak{C}^{\mathfrak{A}} \subseteq \mathfrak{C}^{\mathfrak{B}}$ then $\mathfrak{A} \subseteq \mathfrak{B}$ (there are provable counterexamples, e.g. take $\mathfrak{A} = \mathfrak{C} = \mathbf{PSPACE}$ and $\mathfrak{B} = \mathbf{LogSPACE}$).

1.2.1 Lowness

It is not always true that giving a complexity class an oracle changes it. For example, if $L \in \mathbf{P}$, then $\mathbf{P}^L = \mathbf{P}$, intuitively because $\mathbf{P}$ algorithms can call other $\mathbf{P}$ algorithms as subroutines. We say that L *is low for* $\mathbf{P}$. As another example, the language GraphIsomorphism is low for $\mathbf{NP}$ and for $\mathbf{PP}$ [22]. Intuitively, a language being low for some class indicates that it is not nearly as complex as the languages in that class, because even instant access to solutions provides no help to solve new problems.

Counterintuitively, it is not true that if L is low for $\mathfrak{C}$ and $\mathfrak{C} \subseteq \mathfrak{D}$ then L is low for $\mathfrak{D}$, even though $\mathfrak{D}$ contains more difficult problems than $\mathfrak{C}$. This breaks a little bit with the idea that if $\mathfrak{C} \subseteq \mathfrak{D}$ then the problems of $\mathfrak{D}$ are more difficult than those in $\mathfrak{C}$. For example, we know that $\mathbf{NP} \subseteq \mathbf{PP}$ and that $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is low for $\mathbf{NP}$, but we do not know whether $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is low for $\mathbf{PP}$ (Here $\mathbf{PP}$ is the class of problems solvable by polynomial-time Turing Machines which are allowed to flip random coins but which must give the right answers with higher probability than wrong answers. The language MajoritySAT is complete for this class). Why this discrepancy in lowness? At a high level, it is because these two classes are defined with different Turing Machine models: the former according to the non-deterministic model, the latter the probabilistic model. A language is in $\mathbf{NP}$ if a solution can be easily checked, whereas a language is in $\mathbf{PP}$ if there is a good way to control the number of accepting computation paths of a non-deterministic machine that accepts it. These two concepts do not carry over one-to-one.

This pair of classes, $\mathbf{NP}$ and $\mathbf{PP}$, is a good example of where difficulty as gauged by time consumption by algorithms and by reductions is very different. The problems SAT and MajoritySAT both take $2^{O(n)}$ time by today's best algorithms, yet SAT $\leq_m^p$ MajoritySAT and it is not obvious that MajoritySAT $\leq_m^p$ SAT, in fact the former statement relativizes, but the latter statement does not. Hence $\mathbf{NP}$ is usually thought of as a smaller, much less powerful class than $\mathbf{PP}$ in terms of computational complexity.

1.3 Definitions and table of nomenclature

If $x \in \{0,1\}^n$ is a string of bits, then by $|x\rangle_n$ we mean the x-th basis vector in $\mathbb{C}^{2^n}$:

$$|x\rangle_n \equiv [\underbrace{0\ 0\cdots 0}_{x}\ 1\ \underbrace{0\ 0\cdots 0}_{2^n-x-1}]^T \tag{1.5}$$

The mapping from strings to numbers is the natural one, so the string 000

represents the number 0, the string 011 represents the number 3, etc. Here a state's subscript denotes its size, that is, the number of qubits. For example, $|x\rangle_n$ consists of n qubits. Sometimes the subscript is omitted because the size of the register is immaterial, or obvious from context. We denote by $\mathcal{B} \equiv \{z \in \mathbb{C}^2 \mid |z| = 1\}$ the state space of a qubit, and by $\mathcal{B}^{\otimes n} \equiv \{z \in \mathbb{C}^{2^n} \mid |z| = 1\}$ the state space of n qubits.

The letter x will always refer to the input string to the current machine, and n is reserved to refer to its length, $n = \text{length}(x)$. Whenever a mention of polynomial time, or polynomial space, is made, it is understood that we mean that the function is polynomial as a function of n. The letters L and K always refer to languages, i.e. $L \subseteq \{0,1\}^*$. When a quantum gate takes an input register and a workspace register, the workspace register is always initialised to $|0\rangle$, and we will often neglect to specify how the operator behaves when the workspace is not properly initialised, as it is clear that the operator can be completed to a unitary operator in many ways.

Definition 1 BQP

A language $L \subseteq \{0,1\}^*$ is in **BQP** if there is a polynomial $w(n)$ and a classical polynomial-time algorithm which on input $x \in \{0,1\}^*$ outputs a description of a quantum circuit U_x. This quantum circuit takes as input only a $w(\text{length}(x))$-qubit register as workspace, initialised to $|0\rangle_w$ (it sometimes convenient to think of U_x as having x "hardwired"), and satisfies the following requirements:

- If $x \in L$ then $\Pr[U_x \text{ accepts } |x\rangle] \geq {}^2/_3$,
- If $x \notin L$ then $\Pr[U_x \text{ accepts } |x\rangle] \leq {}^1/_3$.

Whenever there is an algorithm, such as the one described above, to generate a circuit for each input, the resulting circuit family $\{U_x\}_{x\in\{0,1\}^*}$ is called a polynomial time uniformly generated circuit family, or a uniform circuit family for short. By a counting argument, there are $\aleph_1$ circuit families, but only $\aleph_0$ algorithms, so most circuit families are *non-uniform*. By a similar counting argument, there are $\aleph_1$ Boolean functions $f\colon \{0,1\}^* \to \{0,1\}$, so most functions are not computable by uniform circuits (or by any algorithm, for that matter).

Definition 2 QMA
A language $L \subseteq \{0,1\}^*$ is in **QMA** if there are polynomials $m(n)$, $w(n)$ and a uniform family of quantum circuits $\{U_x\}_{x\in\{0,1\}^*}$, taking an $m(n)$-qubit register as input and a $w(n)$-qubit workspace, such that for all $x \in \{0,1\}^*$:

- **Completeness:** If $x \in L$ then there exists a quantum state $|\psi\rangle_m$ of m qubits such that $\Pr[U \text{ accepts } |\psi\rangle_m] \geq {}^2/_3$,
- **Soundness:** If $x \notin L$ then for all quantum states $|\psi\rangle_m$ on m qubits, $\Pr[U \text{ accepts } |\psi\rangle_m] \leq {}^1/_3$.

The set $\{U_x\}_{x\in\{0,1\}^*}$ of circuits is called the **QMA** protocol for L.

The state $|\psi\rangle_m$ is called the *witness* or the *certificate* for x, similar to how an **NP** Machine accepts a certificate.

In the definitions above, the constants ${}^1/_3$ and ${}^2/_3$ can be replaced by $2^{-\varepsilon(n)}$ and $1-2^{-\varepsilon(n)}$ whenever $\varepsilon(n)$ is a polynomial [62]. This is known as error amplification, or boosting (and is usually achieved simply by executing the protocol a number of times and taking the majority vote among the executions). A circuit with a small probability or error is often called a *boosted* circuit.

The following list of complexity classes is far from exhaustive, and the reader is encouraged to browse the Complexity Zoo [66] for all 535 complexity classes, and to consult a textbook such as Arora and Barak's *Computational Complexity: A Modern Approach*[1] for a comprehensive introduction to computational complexity. All occurrences of n refer to the length of the string x, thus $n = |x| = \text{length}(x)$.

P	Polynomial time	The set of languages accepted by Turing Machines whose running time $t(n)$ is bounded from above by a polynomial: for all $x \in \{0,1\}^*$: $t(n) \leq n^k$.
NP	Nondeterministic polynomial time	The set of languages L for which there is a polynomial-time nondeterministic Turing Machine M such that (i: completeness) if $x \in L$ then $M(x)$ has an accepting computation path and (ii: soundness) if $x \notin L$ then every computation path of $M(x)$ rejects.
PP	Probabilistic polynomial time	Languages such that there is a polynomial-time randomized Turing Machine M such that on all inputs, the probability that $M(x)$ outputs the correct answer is greater than the probabiltiy of obtaining the wrong answer.
BPP	Bounded-error probabilistic polynomial time	Same as **PP** except the machine must output the right answer with probability at least $2/3$.
BQP	Bounded-error quantum polynomial time	Same as **BPP**, except the Turing Machine is a quantum circuit.
MA	Merlin-Arthur	Languages such that there is a randomized polynomial-time Turing Machine M such that (i: completeness) if $x \in L$, then there is a string y such that $M(x,y)$ accepts with high probability and (ii: soundness) if $x \in L$ then $M(x,y)$ rejects with high probability for all strings y.
QCMA	Quantum Merlin, Classical Arthur	Same as **MA** except the machine is a uniform quantum circuit
QMA	Quantum Merlin Arthur	Same as **QCMA** except the witness can be a quantum state.
AM	Arthur Merlin	Like **MA** but Arthur gets to ask Merlin a question, i.e. send a message.
QAM	Quantum Arthur Merlin	Like **AM** but Arthur has a quantum computer. The question is still a classical message.
PSPACE	Polynomial space	Same as **P** except the usage of space, instead of time, is bounded by a polynomial.
PH	Polynomial hierarchy	Let $\Sigma_1^{\mathbf{P}} = \mathbf{NP}$ and $\Sigma_{k+1}^{\mathbf{P}} = \mathbf{NP}^{\Sigma_k^{\mathbf{P}}}$. Then $\mathbf{PH} = \cup_{k=1}^{\infty} \Sigma_k^{\mathbf{P}}$.
CH	Counting hierarchy	Let $\mathbf{C}_1^{\mathbf{P}} = \mathbf{PP}$ and $\mathbf{C}_{k+1}^{\mathbf{P}} = \mathbf{PP}^{\mathbf{C}_k^{\mathbf{P}}}$. Then $\mathbf{CH} = \cup_{k=1}^{\infty} \mathbf{C}_k^{\mathbf{P}}$.
$L \leq_m^p K$	Karp-reducibility	There is a polynomial-time computable function f such that, for all $x \in \{0,1\}^*$, $x \in L \iff f(x) \in K$. We say that L *m-reduces to* K.
$L \leq_T^p K$	Turing-reducibility	There is a polynomial-time oracle Turing Machine with oracle access to K which solves L.
	$K \in \mathfrak{C}$-Complete	Every language in $\mathfrak{C}$ can be reduced to K (under some appropriate notion of reduction, usually mentioned in context), and $K \in \mathfrak{C}$.

The following inclusions are known [1, 58] [3] , and relativize. It is unknown whether any of the inclusions are strict.

Theorem 1. *For all oracles* $\mathcal{O} \subseteq \{0,1\}^*$:

$$\mathbf{P}^{\mathcal{O}} \subseteq \mathbf{NP}^{\mathcal{O}} \subseteq \mathbf{QMA}^{\mathcal{O}} \subseteq \mathbf{PP}^{\mathcal{O}} \subseteq \mathbf{CH}^{\mathcal{O}} \subseteq \mathbf{PSPACE}^{\mathcal{O}} \tag{1.6}$$

[3] We cite [58] for the result **QMA** $\subseteq$ **PP**, because it is the first published proof known to the authors. It is mentioned in the text that defines **QMA** [62], where it is left as an excercise to the reader.

Chapter 2

The quantum polynomial hierarchy

Consider the following problem. The input consists of a graph G and an integer k, and the output should be *"yes"* if and only if the largest clique in G has size k.

It is not clear that this problem is in **NP**: Certainly if G has a clique of size k, then that can be certified by giving the vertices of the clique, but there is no obvious way to certify that there is no larger clique. Moreover, the problem is not easily seen to be in **co-NP** either: if the largest clique of G does not contain exactly k vertices, then that can be verified if it has an even larger clique, but if its largest clique is smaller than k, what then?

However, if we are allowed to call the (**NP**-Complete) language CLIQUE as an oracle, then this puzzle is easily solved: first we query CLIQUE on $\langle G, k\rangle$ to see whether G has *any* clique of size k, and then we query CLIQUE on $\langle G, k+1\rangle$ to see whether k is the size of the largest clique, or not.

The polynomial hierarchy neatly classifies a host of problems such as these, which do not seem to be captured by **NP**-Completeness. The example above can be solved by a polynomial-time, deterministic oracle Turing Machine with access to an **NP** language, but one may also imagine giving a *non*-deterministic Turing Machine an **NP** oracle. Repeating this procedure recursively gives us the various levels of the polynomial hierarchy: The first level consists of just **P** and **NP**, the second level is $\mathbf{P}^{\mathbf{NP}}$ and $\mathbf{NP}^{\mathbf{NP}}$, which we call Δ_2^P and Σ_2^P, and the third level, called Σ_3^P, is **NP** with an oracle for a Σ_2^P language, denoted $\mathbf{NP}^{\Sigma_2^P}$. This hierarchy was first defined in [3] by Meyer and Stockmeyer.

Let's see two more examples motivating the study of the polynomial hierarchy.

First, each of the levels of the hierarchy has natural complete problems. For example, the problem of deciding, given a Boolean formula ϕ, whether there *exists* a y such that *for all* z: $\phi(y, z)$ is true, is the archetypical Σ_2^P-Complete problem. Second, in 1983, Sipser [12] showed how any language in the class **BPP** can be rewritten this way, as a predicate with two quantifiers, showing that $\mathbf{BPP} \subseteq \Sigma_2^P$. Hence the two ostensibly unrelated concepts of nondeterminism and randomness are brought together by studying the polynomial hierarchy.

Motivated by the fruits of the study of this classical object, we define a quantum version by analogy. In this analogy, **P** corresponds to **BQP**, **NP** corresponds to **QMA**, and subsequent levels of the hierarchy are defined recursively as before. We call the resulting hierarchy **BQPH**:

$$\Sigma_1^{\mathbf{BQP}} = \mathbf{QMA} \qquad \Sigma_{i+1}^{\mathbf{BQP}} = \mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} \tag{2.1}$$

$$\Delta_1^{\mathbf{BQP}} = \mathbf{BQP} \qquad \Delta_{i+1}^{\mathbf{BQP}} = \mathbf{BQP}^{\Sigma_i^{\mathbf{BQP}}} \tag{2.2}$$

$$\mathbf{BQPH} = \bigcup_{i=1}^{\infty} \Sigma_i^{\mathbf{BQP}} \tag{2.3}$$

2.1 Properties of the quantum polynomial hierarchy

A fundamental property of the polynomial hierarchy is that either it is infinite, or else it "collapses" to some finite level. By infinite, we mean that each level is a strict subset of the next: $\mathbf{NP} \subsetneq \Sigma_2^{\mathbf{P}} \subsetneq \Sigma_3^{\mathbf{P}} \subsetneq \cdots$. By "collapse", we mean that for some i, $\Sigma_i^{\mathbf{P}} = \Sigma_k^{\mathbf{P}}$ for all $k \geq i$ and hence $\Sigma_i^{\mathbf{P}} = \mathbf{PH}$. We say that the hierarchy has collapsed to the i-th level. In a sense, this is the only way it can collapse: it cannot be that $\Sigma_i^P = \Sigma_{i+1}^P \neq \Sigma_{i+2}^P$. Because many researchers believe that the polynomial hierarchy does not collapse, showing that a hypothesis implies a collapse has become a common way of giving evidence against a hypothesis.

We will show that the quantum polynomial hierarchy collapses in the same way under analogous hypotheses. The following theorem will be the principal ingredient.

Theorem 2. *For every oracle set* $\mathcal{O} \subseteq \{0,1\}^\star$, $\mathbf{QMA}^{\mathbf{BQP}^{\mathcal{O}}} = \mathbf{QMA}^{\mathcal{O}}$, *i.e.* $\mathbf{BQP}^{\mathcal{O}}$ *is low for* $\mathbf{QMA}^{\mathcal{O}}$.

In other words, the statement that **BQP** is low for **QMA** relativizes.

2.1.1 Conditional collapses

The simplest and earliest conditional collapse of the classical polynomial hierarchy is the following: if $\mathbf{P} = \mathbf{NP}$ then the polynomial hierarchy collapses to $\mathbf{PH} = \mathbf{P}$ [3]. This theorem is true for our hierarchy, too:

Theorem 3. *If* $\mathbf{BQP} = \mathbf{QMA}$ *then* $\mathbf{BQPH} = \mathbf{BQP}$.

Proof. As a warm-up, let's first prove that $\mathbf{QMA}^{\mathbf{QMA}} = \mathbf{BQP}$ under the assumption that $\mathbf{BQP} = \mathbf{QMA}$. By hypothesis, we get $\mathbf{QMA}^{\mathbf{QMA}} = \mathbf{QMA}^{\mathbf{BQP}}$. But we know from Theorem 2 that $\mathbf{BQP}$ is low for $\mathbf{QMA}$, i.e. $\mathbf{QMA}^{\mathbf{BQP}} = \mathbf{QMA}$. Hence the case $\Sigma_2^{\mathbf{BQP}} = \mathbf{BQP}$ is proved. In list format:

$$
\begin{aligned}
\mathbf{BQP} &= \mathbf{QMA} && \text{The assumption} && (2.4)\\
\Sigma_2^{\mathbf{BQP}} &= \mathbf{QMA}^{\mathbf{QMA}} && \text{By definition} && (2.5)\\
&= \mathbf{QMA}^{\mathbf{BQP}} && \text{By assumption} && (2.6)\\
&= \mathbf{QMA} && \mathbf{BQP} \text{ is low for } \mathbf{QMA} && (2.7)\\
&= \mathbf{BQP} && \text{By assumption} && (2.8)
\end{aligned}
$$

For higher levels of the hierarchy, the proof is by induction. The base case is $\Sigma_1^{\mathbf{BQP}} = \mathbf{BQP}$. For the induction case, we suppose that $\Sigma_i^{\mathbf{BQP}} = \mathbf{BQP}$ for some i and will derive that $\Sigma_{i+1}^{\mathbf{BQP}} = \mathbf{BQP}$.

By definition, $\Sigma_{i+1}^{\mathbf{BQP}} = \mathbf{QMA}^{\Sigma_i^{BQP}}$. Substituting the induction hypothesis, we get $\mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} = \mathbf{QMA}^{\mathbf{BQP}} = \mathbf{BQP}$. In list format:

$$
\begin{aligned}
\Sigma_i^{\mathbf{BQP}} &= \mathbf{BQP} && \text{The induction hypothesis} && (2.9)\\
\Sigma_{i+1}^{\mathbf{BQP}} &= \mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} && \text{By definition} && (2.10)\\
&= \mathbf{QMA}^{\mathbf{BQP}} && \text{By the induction hypothesis} && (2.11)\\
&= \mathbf{QMA} && \mathbf{BQP} \text{ is low for } \mathbf{QMA} && (2.12)\\
&= \mathbf{BQP} && \text{By assumption} && (2.13)
\end{aligned}
$$

For every i, the class $\Sigma_i^{\mathbf{BQP}}$ collapses to $\mathbf{BQP}$, which we substitute in the definition of $\mathbf{BQPH}$ to complete the proof:

$$\mathbf{BQPH} = \bigcup_{i=1}^{\infty} \Sigma_i^{\mathbf{BQP}} = \bigcup_{i=1}^{\infty} \mathbf{BQP} = \mathbf{BQP} \tag{2.14}$$

□

We are now ready to proceed to the next theorem, which generalizes Theorem 3 to other levels of the hierarchy:

Theorem 4. *If* $\Delta_i^{\mathbf{BQP}} = \Sigma_i^{\mathbf{BQP}}$*, then* $\mathbf{BQPH} = \Delta_i^{\mathbf{BQP}}$.

Proof. The proof is by induction. Take i to be some positive integer. The case of $i = 1$ is the case where $\mathbf{BQP} = \mathbf{QMA}$ and is covered in Theorem 3. For $i > 1$, the base case of the induction is if $\Delta_i^{\mathbf{BQP}} = \Sigma_i^{\mathbf{BQP}}$ then $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+1}^{\mathbf{BQP}}$. The induction step will be that if $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+k}^{\mathbf{BQP}}$ then $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+k+1}^{\mathbf{BQP}}$. First, the base case in list format:

$$\begin{aligned}
\Delta_i^{\mathbf{BQP}} &= \Sigma_i^{\mathbf{BQP}} && \text{The assumption} && (2.15)\\
\Sigma_{i+1}^{\mathbf{BQP}} &= \mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} && \text{By definition} && (2.16)\\
&= \mathbf{QMA}^{\Delta_i^{\mathbf{BQP}}} && \text{By assumption} && (2.17)\\
&= \mathbf{QMA}^{\mathbf{BQP}^{\Sigma_{i-1}^{\mathbf{BQP}}}} && \text{By definition of } \Delta_i^{\mathbf{BQP}} && (2.18)\\
&= \mathbf{QMA}^{\Sigma_{i-1}^{\mathbf{BQP}}} && \mathbf{BQP}\text{is low for } \mathbf{QMA} && (2.19)\\
&= \Sigma_i^{\mathbf{BQP}} && \text{By definition} && (2.20)\\
&= \Delta_i^{\mathbf{BQP}} && \text{By hypothesis} && (2.21)
\end{aligned}$$

The induction case is similar. The induction hypothesis is that $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+k}^{\mathbf{BQP}}$ for some $k \geq 1$, and will imply that $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+k+1}^{\mathbf{BQP}}$.

$$\begin{aligned}
\Delta_i^{\mathbf{BQP}} &= \Sigma_{i+k}^{\mathbf{BQP}}, k \geq 1 && \text{Induction hypothesis} && (2.22)\\
\Sigma_{i+k+1}^{\mathbf{BQP}} &= \mathbf{QMA}^{\Sigma_{i+k}^{\mathbf{BQP}}} && \text{By definition} && (2.23)\\
&= \mathbf{QMA}^{\Delta_i^{\mathbf{BQP}}} && \text{By induction hypothesis} && (2.24)\\
&= \mathbf{QMA}^{\mathbf{BQP}^{\Sigma_{i-1}^{\mathbf{BQP}}}} && \text{By definition} && (2.25)\\
&= \mathbf{QMA}^{\Sigma_{i-1}^{\mathbf{BQP}}} && \mathbf{BQP} \text{ is low for } \mathbf{QMA} && (2.26)\\
&= \Sigma_i^{\mathbf{BQP}} && \text{By definition} && (2.27)\\
&= \Delta_i^{\mathbf{BQP}} && \text{By assumption} && (2.28)
\end{aligned}$$

So for every $k \geq 0$, $\Delta_i^{\mathbf{BQP}} = \Sigma_{i+k}^{\mathbf{BQP}}$, so $\mathbf{BQPH} = \Delta_i^{\mathbf{BQP}}$. □

The classical hierarchy has one more elementary conditional collapse, namely if any level is closed under complement, then it collapses to that level, e.g. if $\mathbf{NP} = \mathbf{coNP}$ then $\mathbf{NP} = \mathbf{PH}$. In the quantum case, we do not presently know how to prove the analogous statement, i.e. if $\mathbf{QMA} = \mathbf{coQMA}$ then $\mathbf{QMA} = \mathbf{BQPH}$. The principal difficulty seems to lie in showing that $\mathbf{QMA}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA}$, from which the desired results would follow immediately, or indeed $\mathbf{BQP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA}$, from which the desired collapse would not immediately follow. We do, later on, prove something almost as good, namely $\mathbf{P}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA}$ (Theorem 12), which we use to give a novel, very short proof of a theorem by Vyalyi (Theorem 16).

There are many other less elementary conditional collapses. For example, the Karp-Lipton Theorem [21] states that the hierarchy collapses to $\mathbf{PH} = \Sigma_2^P$ if $\mathbf{NP} \subset \mathbf{P}_{/poly}$, and we will return to quantum analogues of it in Section 2.6.

2.1.2 Upper bounds on the quantum polynomial hierarchy

We present one upper bound on the quantum polynomial hierarchy, namely that it is contained in the counting hierarchy. This follows from the fact that the inclusion $\mathbf{QMA} \subseteq \mathbf{PP}$ relativizes [9].

Theorem 5. *Our hierarchy is in the counting hierarchy:* $\mathbf{BQPH} \subseteq \mathbf{CH}$. *In fact, for each* i, $\Sigma_i^{\mathbf{BQP}} \subseteq \mathbf{C}_i^{\mathbf{P}}$.

Proof. Recall that the counting hierarchy, $\mathbf{CH}$, is defined as $\mathbf{C}_1^{\mathbf{P}} = \mathbf{PP}$ and $\mathbf{C}_{i+1}^{\mathbf{P}} = \mathbf{C}_i^{\mathbf{PP}}$ with $\mathbf{CH} = \bigcup_{i=1}^{\infty} \mathbf{C}_i^{\mathbf{P}}$.

The proof is by induction. In the base case, we have $\Sigma_1^{\mathbf{BQP}} = \mathbf{QMA} \subseteq \mathbf{PP} = \mathbf{C}_1^{\mathbf{P}}$, which was shown by Marriot and Watrous [9]. In the induction step, we will assume that $\Sigma_i^{\mathbf{BQP}} \subseteq \mathbf{C}_i^{\mathbf{P}}$ and derive $\Sigma_{i+1}^{\mathbf{BQP}} \subseteq \mathbf{C}_{i+1}^{\mathbf{P}}$.

$$
\begin{aligned}
\text{Suppose } \Sigma_i^{\mathbf{BQP}} &\subseteq \mathbf{C}_i^{\mathbf{P}}, \text{ for } i \geq 1 && \text{Induction hypothesis} && (2.29)\\
\Sigma_{i+1}^{\mathbf{BQP}} &= \mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} && \text{By definition} && (2.30)\\
&\subseteq \mathbf{QMA}^{\mathbf{C}_i^{\mathbf{P}}} && \text{Induction hypothesis} && (2.31)\\
&\subseteq \mathbf{PP}^{\mathbf{C}_i^{\mathbf{P}}} && \mathbf{QMA} \subseteq \mathbf{PP} \text{ relativizes} && (2.32)\\
&= \mathbf{C}_{i+1}^{\mathbf{P}} && \text{By definition} && (2.33)
\end{aligned}
$$

Concluding, by induction: for all i, $\Sigma_i^{\mathbf{BQP}} \subseteq \mathbf{C}_i^{\mathbf{P}}$, and hence $\mathbf{BQPH} \subseteq \mathbf{CH}$. □

The proof works for any complexity class $\mathfrak{C}$ which relativizingly contains **QMA**, and where it makes sense to define a hierarchy $\Sigma_{i+1}^{\mathfrak{C}} = \mathfrak{C}^{\Sigma_i^{\mathfrak{C}}}$. So one could take $\mathfrak{C} = A_0PP$, which is defined by Vyalyi in [58], for example.

Because the counting hierarchy is contained in **PSPACE**, the following is immediate.

Theorem 6. BQPH $\subseteq$ PSPACE.

We have given only *upper* bounds on $\Sigma_i^{\mathbf{BQP}}$. It would be interesting to have *lower bounds*, of the form $\mathbf{QAM} \subseteq \Sigma_3^{\mathbf{BQP}}$, for example. As it stands now, it may still be that the counting hierarchy is infinite, yet **BQPH** collapses to **BQP** and still $\mathbf{BQP} \subsetneq \mathbf{PP}$ and $\mathbf{BQP} \subsetneq \mathbf{QAM}$.

2.2 Closure properties of QMA

This section contains the proof that **QMA** is closed under deterministic polynomial-time Turing reductions, which is the main ingredient to the simpler proof of Vyalyi's Theorem in Section 2.3. As a warm-up, we first show that **QMA** is closed under intersection. The main ingredient is Theorem 8, which tells us about situations in which two subcircuits act independently and hence we do not have to worry about destructive interference. To streamline its proof, we recall the fact that every state has a Schmidt Decomposition.

Theorem 7 (Schmidt Decomposition)**.** *For every pure state $|\psi\rangle \in A \otimes B$ there are orthonormal sets $\{|a_i\rangle\} \subset A$, $\{|b_i\rangle\} \subset B$ such that*

$$|\psi\rangle = \sum_i y_i |a_i\rangle |b_i\rangle \tag{2.34}$$

where y_i are all nonnegative real numbers.

The next lemma, the Entanglement Independence Lemma, says that if two **QMA** protocols possess soundness individually, that property is preserved when they are combined in a single circuit and their inputs are entangled, as long as they are implemented and measured independently, as in Figure 2.1. This is not obvious a priori, because oftentimes quantum circuits behave in surprising ways when clever use is made of entanglement. The result of the Theorem, then, is that in this case, no such clever use is possible for Merlin: entangling the two certificates for the two circuits gives him no advantage. We give two proofs of this theorem.

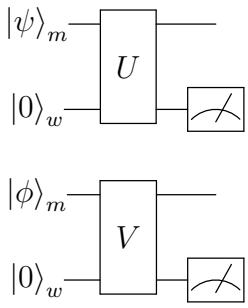

Figure 2.1: The circuits U and V receive inputs that may be entangled.

Theorem 8 (Entanglement Independence Lemma)**.** *Let U, V be two quantum circuits as in Figure 2.1, both receiving an m-qubit input and a w-qubit workspace. Suppose that U and V accept with probability at most a and b, respectively, regardless of their m-qubit input. Then the probability that both U and V accept when they are implemented jointly, and when their inputs may be entangled, is at most $a \cdot b$.*

We will first prove that the probability is at most a, using simple algebraic manipulations.

First proof of Theorem 8. Let Π_U and Π_V denote the measurement operators that correspond to the events that U and V accept. In general, the probability that a system $|\psi\rangle$ is measured according to the event corresponding to Π_A and then Π_B is $\|\Pi_B \Pi_A |\psi\rangle\|^2 = \langle\psi| \Pi_A \Pi_B \Pi_A |\psi\rangle$. In our case, the measurement operators are $\Pi_U \otimes I$ and $I \otimes \Pi_V$. These operators commute, so it makes no difference whether one or the other comes "first".

If the input is the state $|\psi\rangle = \sum_i z_i |a_i\rangle |b_i\rangle$, then the probability that both U and V accept this input is

$$P = \left| (\Pi_U \otimes \Pi_V) \cdot (U \otimes V) \cdot \sum_i z_i |a_i\rangle |0\rangle |b_i\rangle |0\rangle \right|^2 \tag{2.35}$$

$$= \left| \sum_i z_i (\Pi_U U |a_i\rangle |0\rangle) \otimes (\Pi_V V |b_i\rangle |0\rangle) \right|^2 \tag{2.36}$$

$$\leq \left| \sum_i z_i \Pi_U U |a_i\rangle |0\rangle \right|^2 = \Pr\left[U \text{ accepts } \sum_i z_i |a_i\rangle \right] \leq a \tag{2.37}$$

□

The next proof uses a trick by Marriot and Watrous [9]. They are able to express the probability that a circuit accepts an m-qubit input in terms of the eigenvalues of a $2^m \times 2^m$ matrix, as in Equation 2.39. The equality follows from the fact that$(I^{\otimes m} \otimes |0\rangle)(|\psi\rangle_m) = |\psi\rangle_m \otimes |0\rangle$.

$$P[U \text{ accepts } |\psi\rangle] = \langle\psi| \langle 0| U^\dagger \Pi U |\psi\rangle |0\rangle \tag{2.38}$$

$$= \langle\psi| \cdot \left((I^{\otimes m} \otimes \langle 0|) \cdot U^\dagger \Pi U \cdot (I^{\otimes m} \otimes |0\rangle) \right) |\psi\rangle \tag{2.39}$$

Second proof of Theorem 8. Let

$$\tilde{U} = (I^{\otimes m} \otimes \langle 0|) \cdot U^\dagger \Pi U \cdot (I^{\otimes m} \otimes |0\rangle) \tag{2.40}$$

and define $\tilde{V}$ analogously. Clearly these operators are Hermitian, so their eigenvalues are real, so their "largest" eigenvalue is well-defined.

According to Equation 2.39, for any m+m-qubit quantum state $|\phi\rangle = \sum_i y_i |a_i\rangle |b_i\rangle$, the probability that U and V both accept is $\langle\phi| (\tilde{U} \otimes \tilde{V}) |\phi\rangle$. This probability is maximized at the largest eigenvalue of $\tilde{U} \otimes \tilde{V}$. But the eigenvalues of $\tilde{U} \otimes \tilde{V}$ are exactly the products of the eigenvalues of $\tilde{U}$ and $\tilde{V}$. More precisely, for every pair of eigenvalues λ, μ of U and V, $\lambda \cdot \mu$ is an eigenvalue of $\tilde{U} \otimes \tilde{V}$.

By assumption, the largest eigenvalues of U and V are at most a and b, so the largest eigenvalue of $\tilde{U} \otimes \tilde{V}$ is at most $a \cdot b$. □

The approach of Marriot and Watrous has the advantage that the qubits of the workspace are encapsulated by the operator $\tilde{U}$, which allows us to express acceptance of U, and in turn express perfect play by Merlin, in terms of the eigenvalues of $\tilde{U}$:

$$P[U \text{ accepts } \mid \text{ Perfect play by Merlin}] = \max_{|\psi\rangle} \langle\psi| \tilde{U} |\psi\rangle \tag{2.41}$$

This quantity is maximized at an eigenvalue of $\tilde{U}$, and the message that Merlin will send to maximize Arthur's probability of acceptance is a corresponding eigenvector. The second proof of Theorem 8 shows that if the circuit consists of multiple non-interacting parts, then one of the maximizing eigenvectors is an unentangled state. The upshot, then, is exactly what we wanted: Merlin has the ability to entangle his certificates, but he gains no advantage from doing so, because Arthur's acceptance probability is maximized at an unentangled certificate.

The next theorem tells us that if some event happens with overwhelming probability (more precisely, with only exponentially small bias), then the probability

that the event happens a polynomial number of times in a row is also overwhelming.

Theorem 9. *If $\{X_i\}$ are n^k independent identically distributed random binary variables with $P[X_i = 1] \geq 1 - e^{-n}$ and k a positive constant, then the probability that $X_i = 1$ for all i tends to 1 as $n \to \infty$.*

Proof. First we express the probability that all $X_i = 1$.

$$P(n) = \Pr\left[\bigwedge_{i=1}^{n^k} X_i = 1\right] = (\Pr[X_1 = 1])^{n^k} = \left(1 - e^{-n}\right)^{n^k} \tag{2.42}$$

Recall the following limit.

$$\lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \tag{2.43}$$

We simply substitute our probability, which is $P(n) = 1 - e^{-n}$, and our number of trials, which is n^k:

$$\lim_{n\to\infty} P(n) = \lim_{n\to\infty} \left(\left(1 - \frac{1}{e^n}\right)^{e^n}\right)^{\frac{n^k}{e^n}} = \lim_{n\to\infty} \left(\frac{1}{e}\right)^{\frac{n^k}{e^n}} = \lim_{n\to\infty} e^{-\frac{n^k}{e^n}} \tag{2.44}$$

Now we note that $\frac{n^k}{e^n}$ tends to 0 as $n \to \infty$, which completes the proof:

$$\lim_{n\to\infty} \Pr\left[\bigwedge_{i=0}^{n^k} X_i = 1\right] = e^0 = 1 \tag{2.45}$$

□

In our setting, the random variable is the probability that a **QMA** circuit gives the right output.

From here, it is easy and fun to derive the facts that **QMA** is closed under (i) union and (ii) intersection, and (iii) that **QMA**$\cap$**co-QMA** is closed under Turing reductions. To see why **QMA** is closed under intersection, consider the circuit in Figure 2.2: one simply asks Merlin for certificates to both problems, measures, and accepts iff both of the circuits accept.

Theorem 10. QMA *is closed under intersection.*

Proof. Let $A \in \mathbf{QMA}$ and $B \in \mathbf{QMA}$. For a particular input string $x \in \{0,1\}^*$, we now design a simple **QMA** protocol for membership in $A \cap B$. Let U and V be the circuits of the **QMA** protocols for the languages A and B, respectively. We ask Merlin for two certificates $|\psi_A\rangle_p$ and $|\psi_B\rangle_q$, feed them to the circuits and measure the outcomes, like in Figure 2.2. We accept iff both circuits accept. The difficulty is that Merlin can entangle the two certificates, but we will show that he gains no advantage by doing so.

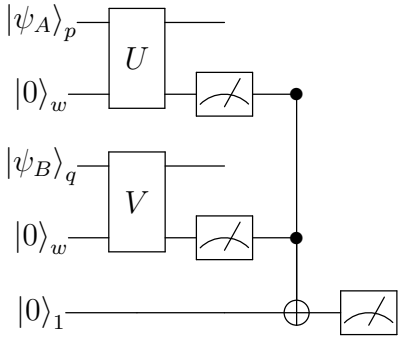

Figure 2.2: A circuit which receives two certificates $|\psi_A\rangle_p$ and $|\psi_B\rangle_q$, possibly entangled, and executes the protocols for languages A and B on them.

Part 1: Completeness. Suppose that $x \in A \cap B$. Then Merlin can be honest and send us the state $|\psi_A\rangle_p \otimes |\psi_B\rangle_q$. These are two unentangled certificates for membership of A and B, so the two subcircuits U and V can be analyzed independently: U will accept with probability $|\Pi U |\psi_A\rangle| \geq {}^9/_{10}$ and V will accept with probability $|\Pi V |\psi_B\rangle \geq {}^9/_{10}$. Therefore both circuits accept with probability at least $({}^9/_{10})^2 \geq {}^2/_3$, which suffices to show that the protocol has completeness.

Part 2: Soundness. Suppose that $x \notin A \cap B$, e.g. because $x \notin A$. Then Merlin may send us any arbitrary, possibly highly entangled state. If U is implemented alone, then it accepts any certificate with probability at most ${}^1/_3$. Here, we provide U with a well-initialised workspace of $|0\rangle_w$, so by the Entanglement Independence Lemma, U will also accept with low probability, at most ${}^1/_3$, regardless of how Merlin entangles its input register with the rest of the certificate. Therefore the probability that the circuit as a whole accepts is also $\leq {}^1/_3$, so the protocol has soundness. □

As a warm-up for the proof that **QMA** is closed under Turing reductions, we recall why **NP** has that property.

Theorem 11. $\mathbf{NP} \cap \mathbf{coNP}$ *is closed under deterministic Turing reductions. That is,* $\mathbf{P}^{\mathbf{NP} \cap \mathbf{coNP}} = \mathbf{NP} \cap \mathbf{coNP}$.

Proof. The trivial direction is $\mathbf{NP} \cap \mathbf{co\text{-}NP} \subseteq \mathbf{P}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}}$, so we will only show the other direction, $\mathbf{P}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$. To this end, it is sufficient to show that $\mathbf{P}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} \subseteq \mathbf{NP}$ because $\mathbf{P}$ is closed under complement.

Let $L \in \mathbf{NP} \cap \mathbf{co\text{-}NP}$ be a language and M^L a polynomial-time (say $t(n)$-time, with $n = |x| = \text{length}(x)$) deterministic Turing Machine with oracle access to L. Since $L \in \mathbf{NP} \cap \mathbf{co\text{-}NP}$, there are non-deterministic Turing Machines Y and N recognizing the languages L and $\overline{L}$, respectively, both running in polynomial time (say $t'(n)$ time). If we manage to simulate M^L with a non-deterministic machine, we will have proved the theorem.

Clearly *if* we manage to obtain the answers to all the oracle queries M^L makes, *then* we can faithfully simulate M^L and obtain the right answer. The central insight is that we can obtain the answers by guessing and then verifying them. Therefore the algorithm will be as follows.

Before we compute anything, (i) we non-deterministically guess the answers $a_1, \ldots, a_{t(n)}$ to all the queries that M^L makes, (ii) we guess that the queries that M^L is going to make are the strings $s_1, \ldots, s_{t(n)}$, and lastly (iii) we guess certificate strings $y_1, \ldots, y_{t(n)}$ and $z_1, \ldots, z_{t(n)}$ for the machines Y and N, respectively.[1] The remainder of the computation is deterministic. The algorithm checks that its guesses were correct: for each i, it checks that $Y(s_i, y_i) = a_i$ and $N(s_i, z_i) = \neg a_i$ (meaning that, for example, if $a_2 = 1$ then Y accepts the certificate y_2 we guessed for s_2 and N rejects z_2). If any of these checks fail, we have evidently guessed incorrectly, and we reject.

Lastly, we simulate M^L. When it makes the i-th query, we check that it queries $s_i \stackrel{?}{\in} L$; if so, we feed it the answer a_i and continue the simulation, but if not, we immediately reject, because we have incorrectly guessed which strings M^L would query. When M^L halts and accepts, we accept; otherwise we reject.

If M^L accepts, then our machine accepts. The accepting paths are exactly those that correctly guessed which strings M^L was going to query, what the answers were, and what certificates would satisfy Y and N. If M^L rejects, then our machine rejects too, because even the paths that obtained correct answers for the oracle queries still reject when M^L halts and rejects. □

At long last, we are ready to prove that **QMA** is closed under Turing reductions.

[1] The strings s_i are not longer than $t(n)$ bits, because M^L runs in time $t(n)$ bits. Therefore the certificates y_i are not longer than $t'(t(n))$ bits.

Theorem 12. *The class* **QMA∩co-QMA** *is closed under deterministic polynomial-time Turing reductions. That is,* $\mathbf{QMA} \cap \mathbf{co\text{-}QMA} = \mathbf{P}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}}$.

Proof. By the observations in Theorem 11, it suffices to give a **QMA** protocol for a language in $\mathbf{P}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}}$. So let $L \in \mathbf{QMA} \cap \mathbf{co\text{-}QMA}$ and let M^L be a deterministic polynomial-time (say $t(n)$-time) Turing Machine with oracle access to L, and let $\{Y_s\}$ and $\{N_s\}$ be the circuit families corresponding to the **QMA** protocols for L and $\overline{L}$, respectively.

In the previous proof, we fed M^L truthful answers to all its oracle queries. Here we will settle for something which suffices for our purposes: either (i) if $x \in L$, then with very high probability we will give M^L truthful answers to its queries, or else (ii) if $x \notin L$, with very high probability we will detect any attempt of Merlin to fool us, and reject. To this end, we assume that Y_s and N_s are amplified circuits such that if $s \in L$, then there is a state that Y_s accepts with probability $\geq 1 - 2^{-n}$, whereas if $s \notin L$, then Y_s rejects every state with probability $\geq 1 - 2^{-n}$.

In this protocol, we expect that Merlin's message contains (i) answers to the queries that M^L asks, (ii) which (classical) strings the machine queries and (iii) quantum certificate states $|\psi_1\rangle, \ldots, |\psi_t\rangle$ and $|\phi_1\rangle, \ldots, |\phi_t\rangle$. These are the only parts of Merlin's message that need to be quantum, because parts (i) and (ii) are simply classical bit strings.

Before we simulate M^L, we measure all the qubits which we expect to be classical bits (the answers and the query strings) in the computational basis. Then we compute, for each i, whether Y_{s_i} accepts $|\psi_i\rangle$ and whether N_{s_i} accepts $|\phi_i\rangle$. We reject if Y_{s_i} does not output a_i or N_{s_i} does not output $\neg a_i$, just as before. Lastly, we simulate M^L just as in Theorem 11, rejecting if M^L queries a string we did not anticipate or if M^L rejects. Otherwise, we accept.

The only part where we use quantum computing is in the evaluation of the certificates, and we have to argue that this does not influence the soundness of the protocol.

Part 1: Completeness. Suppose that M^L accepts. Then Merlin can be honest and simply send us good, unentangled certificates for the algorithms Y_s and N_s. In particular, for any query s, if $s \in L$, then Y_s will accept the good certificate $|\psi_i\rangle$ with probability $1 - 2^{-n}$ and N_s will reject its certificate $|\phi_i\rangle$ with the same probability. Because these circuits are implemented independently of one another and the state $|\psi_i\rangle$ is not entangled with $|\phi_i\rangle$, these two events are independent, so the probability that both happen is at least $(1 - 2^{-n})^2$. This needs to happen for all queries, of which there are at most $t(n)$. The probability

that all query oracles proceed this way is at least

$$(1 - 2^{-n})^{2t(n)} \tag{2.46}$$

Because $t(n)$ is a polynomial, this quantity tends to 1 for large n, by Theorem 9, meaning that with probability tending to 1, we obtain correct answers for all queries. After that, simulating M^L is a deterministic computation and we copy its answer, so we output the correct answer with probability tending to 1.

Part 2: Soundness. Suppose that M^L rejects. Then Merlin will send us a possibly very complicated entangled state. We start by measuring all the registers which we expect to be classical in the computational basis, so that these registers are now truly classical bits and are not entangled with the rest of the certificate. The certificates to the queries remain quantum. The fact that the various certificates may be entangled with one another presents the principal hurdle in this proof, which we overcome by the Entanglement Independence Lemma.

Suppose that in our simulation we feed M^L only correct answers. Then we will certainly reject. So in order to make us accept, Merlin must make us compute the wrong answer for at least one query, which happens when $s \in L$ but Y_s rejects and N_s accepts. But according to the Entanglement Independence Lemma, and because the protocol N_s possesses soundness, the probability that N_s accepts any certificate state, no matter how the state is entangled with other parts of the circuit that N_s does not touch, is at most e^{-n}. (Our assymetric focus here on the protocol N_s instead of Y_s is because it is easy enough for Merlin to make Y_s reject by sending a bad certificate. In the classical case he might send a non-satisfying assignment to the satisfiability problem, for example).

Above we have described the "good event" that the circuit N_s rejects a single bad certificate. For our simulation to succeed, each query must be a "good event", that is, it must happen $t(n)$ times. By Theorem 9, this probability tends to 1. □

The following corollary is immediate.

Theorem 13. QMA *is closed under complement if and only if it is closed under polynomial-time Turing reductions.*

The next theorem is almost a quantum analogue of the theorem $\mathbf{NP}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} = \mathbf{NP}$. It is an analogue in the sense that **QMA** is substituted for **NP** in three out of four places, and in the sense that it is true for the same reason. It is not an exact analogue because we show inclusion and not equality, and we do not substitute **QMA** for **NP** in one of four places. The reason we cannot go all the way and prove that $\mathbf{QMA}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} = \mathbf{QMA}$, is interesting in itself.

Theorem 14. $\mathbf{NP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA}$.

Proof. We ask Merlin for (i) the certificate for the **NP** machine that we are simulating and (ii) for the queries and their answers similar to how we asked it in Theorem 12. The first thing the **QMA** machine does is measure the certificate in the computational basis. Merlin is supposed to send a classical string, so if he is honest, nothing happens. If he is dishonest, then the certificate has now collapsed to a classical string, and the **NP** machine will reject this if it receives correct answers to its queries.

After measuring the certificate Merlin gave for the **NP** machine, the remaining computation is a simulation of a deterministic Turing Machine which makes queries to $\mathbf{QMA} \cap \mathbf{co\text{-}QMA}$, that is, it is a $\mathbf{P}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}}$ computation. By Theorem 12, this can be simulated in **QMA**. □

In a similar vein,

Theorem 15. $\mathbf{NP}^{\mathbf{QCMA} \cap \mathbf{co\text{-}QCMA}} \subseteq \mathbf{QCMA}$.

2.3 A new proof of a theorem by Vyalyi

Is quantum computing more powerful than classical computing? Towards an answer, researchers have studied how the complexity classes **BQP**and **QMA** relate to traditional classes. Unfortunately, most **QMA**-Complete problems studied thus far are inherently tied to quantum physics, and it is not obvious what classical problems can be solved in this model, nor how **QMA** relates to the landscape of classical complexity classes. Clearly it can solve **NP**, but can it solve the rest of the polynomial hierarchy? Might there be **QMA** protocols for **PP**-Complete problems like MajoritySAT or even for computing the permanent of a matrix? In 2003, Vyalyi showed that there is a surprising connection between these questions:

Theorem 16 (Vyalyi [58])**.** *If* $\mathbf{QMA} = \mathbf{PP}$ *then* $\mathbf{PH} \subseteq \mathbf{QMA}$.

Vyalyi's Theorem says, in other words, that if there is a **QMA** protocol for MajoritySAT, then there are **QMA** protocols for all languages in the polynomial hierarachy. Since it is believed unlikely that $\mathbf{PH} \subseteq \mathbf{QMA}$, this is taken as evidence that $\mathbf{QMA} \neq \mathbf{PP}$.

The proof Vyalyi originally gave defines a new complexity class called A_0PP and uses Gap functions to characterize **QMA** and compare it to A_0PP; then he uses a special version of Toda's Theorem. Here we show that, in fact, there is a

simple proof of this theorem using the simplest version of Toda's Theorem [10] (that $\mathbf{PH} \subseteq \mathbf{P}^{\mathbf{PP}}$) and using only elementary closure properties of the classes mentioned in the statement of the theorem, specifically Theorem 12 and the fact that **PP** is closed under complement [19].

Proof of Theorem 16. The proof has three ingredients: (i) Toda's Theorem, $\mathbf{PH} \subseteq \mathbf{P}^{\mathbf{PP}}$, (ii) The fact that **PP** is closed under complement, i.e. $\mathbf{PP} = \mathbf{coPP}$ and (iii) Theorem 12, $\mathbf{QMA} \cap \mathbf{coQMA} = \mathbf{P}^{\mathbf{QMA} \cap \mathbf{coQMA}}$.

Suppose that $\mathbf{QMA} = \mathbf{PP}$. Then **QMA** is closed under complement, since **PP** is closed under complement, so $\mathbf{PP} = \mathbf{QMA} \cap \mathbf{co\text{-}QMA}$. But we proved earlier that this class is closed under Turing reductions, so **PP** is closed under Turing reductions: $\mathbf{PP} = \mathbf{P}^{\mathbf{PP}}$. From Toda's Theorem, we know that $\mathbf{PH} \subseteq \mathbf{P}^{\mathbf{PP}}$, and we have the desired inclusion. In summary:

$\mathbf{PP} = \mathbf{QMA}$	The assumption	(2.47)
$= \mathbf{QMA} \cap \mathbf{co\text{-}QMA}$	**PP** is closed under complement	(2.48)
$= \mathbf{P}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}}$	$\mathbf{QMA} \cap \mathbf{co\text{-}QMA}$ is closed	(2.49)
	under Turing reductions	(2.50)
$= \mathbf{P}^{\mathbf{PP}}$	The assumption	(2.51)
$\supseteq \mathbf{PH}$	By Toda's Theorem	(2.52)

□

Examining our new proof, we can derive a stronger consequence than Vyalyi did: if there is a **QMA** protocol for MajoritySat, then not only are there **QMA** protocols for every language in the polynomial hierarchy, there is also a **QMA** protocol for the permanent, which is #**P**-Complete [14].

Theorem 17. *If* $\mathbf{QMA} = \mathbf{PP}$ *then* $\mathbf{QMA} = \mathbf{P}^{\#\mathbf{P}}$ *and there is a* **QMA** *protocol for the permanent.*

Proof. We are looking for a **QMA** protocol for the language

$$\text{Permanent} = \{\langle A, s \rangle \mid s \text{ is the permanent of } A\} \tag{2.53}$$

Clearly Permanent $\in \mathbf{P}^{\#\mathbf{P}}$. Inspecting our proof of Theorem 16, specifically line 2.51, we get $\mathbf{QMA} = \mathbf{P}^{\mathbf{PP}}$. We know that $\mathbf{P}^{\mathbf{PP}} = \mathbf{P}^{\#\mathbf{P}}$, so $\mathbf{QMA} = \mathbf{P}^{\#\mathbf{P}}$, so Permanent $\in \mathbf{QMA}$. □

Using Theorem 14, we can strengthen the consequence to $\mathbf{PH}^{\mathbf{PP}} = \mathbf{QMA}$, as in Theorem 18. Note that $\mathbf{P}^{\mathbf{PP}} \subseteq \mathbf{PH}^{\mathbf{PP}}$, so this result supersedes both Theorem 16 and Theorem 17.

Theorem 18. *If* $\mathbf{QMA} = \mathbf{PP}$ *then* $\mathbf{QMA} = \mathbf{PH}^{\mathbf{PP}}$.

Proof. The class $\mathbf{PH}^{\mathbf{PP}}$ is defined as $\Sigma_1^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{NP}^{\mathbf{PP}}$ and $\Sigma_{i+1}^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{NP}^{\Sigma_i^{\mathbf{P}^{\mathbf{PP}}}}$ with $\mathbf{PH}^{\mathbf{PP}} = \bigcup_{i=1}^{\infty} \Sigma_i^{\mathbf{P}^{\mathbf{PP}}}$. In other words, it is as though we define the polynomial hierarchy as usual, except at the base, we give all **NP** machines a **PP** oracle.

The proof is by induction. The base case is that $\Sigma_1^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{QMA}$, and uses (i) that **PP** is closed under complement and (ii) Theorem 14:

$$\Sigma_1^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{NP}^{\mathbf{PP}} = \mathbf{NP}^{\mathbf{QMA}} = \mathbf{NP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA} \tag{2.54}$$

The last inclusion is Theorem 14. The induction step assumes that $\Sigma_i^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{QMA}$ and derives $\Sigma_{i+1}^{\mathbf{P}^{\mathbf{PP}}} = \mathbf{QMA}$:

$$\begin{aligned}
\Sigma_i^{\mathbf{P}^{\mathbf{PP}}} &= \mathbf{QMA} && \text{Induction hypothesis} && (2.55)\\
\Sigma_{i+1}^{\mathbf{P}^{\mathbf{PP}}} &= \mathbf{NP}^{\Sigma_i^{\mathbf{P}^{\mathbf{PP}}}} && \text{By definition} && (2.56)\\
&= \mathbf{NP}^{\mathbf{QMA}} && \text{Induction hypothesis} && (2.57)\\
&= \mathbf{NP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} && \mathbf{QMA} \text{ closed under complement} && (2.58)\\
&\subseteq \mathbf{QMA} && \text{Theorem 14} && (2.59)
\end{aligned}$$

We have shown that $\mathbf{PH}^{\mathbf{PP}} \subseteq \mathbf{QMA}$, i.e. relative to a **PP** oracle, every level of the polynomial hierarchy is contained in the unrelativized version of **QMA**. For the opposite inclusion, we note that $\mathbf{QMA} \subseteq \mathbf{PP} \subseteq \mathbf{P}^{\mathbf{PP}} \subseteq \mathbf{PH}^{\mathbf{PP}}$ is known unconditionally. □

The techniques we developed allow us to improve Theorem 16 in another direction: we can weaken the hypothesis from $\mathbf{QMA} = \mathbf{PP}$ to $\mathbf{QMA} = \mathbf{co\text{-}QMA}$. This hypothesis is decidedly weaker, because it is conceivable that **QMA** is closed under complement yet is properly contained in **PP**. For this reason, we do not obtain a result similar to Theorem 17, as the permanent problem is only more difficult than MajoritySAT.

Theorem 19. *If* $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ *then* $\mathbf{PH} \subseteq \mathbf{QMA}$.

Proof. The proof is by induction. The base case, $\Sigma_1^{\mathbf{P}} \subseteq \mathbf{QMA}$, is elementary and is true unconditionally. In the induction step, we assume that $\Sigma_i^{\mathbf{P}} \subseteq \mathbf{QMA}$ and derive $\Sigma_{i+1}^{\mathbf{P}} \subseteq \mathbf{QMA}$:

$$\begin{aligned}
\mathbf{QMA} =& \mathbf{QMA} \cap \mathbf{coQMA} && \text{The assumption} && (2.60)\\
\Sigma_i^{\mathbf{P}} \subseteq& \mathbf{QMA} && \text{Induction hypothesis} && (2.61)\\
\Sigma_{i+1}^{\mathbf{P}} =& \mathbf{NP}^{\Sigma_i^{\mathbf{P}}} && \text{Definition} && (2.62)\\
\subseteq& \mathbf{NP}^{\mathbf{QMA}} && \text{By the induction hypothesis} && (2.63)\\
=& \mathbf{NP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} && \text{By assumption} && (2.64)\\
\subseteq& \mathbf{QMA} && \text{By Theorem 14} && (2.65)
\end{aligned}$$

Hence for every i, we have $\Sigma_i^{\mathbf{P}} \subseteq \mathbf{QMA}$, so $\mathbf{PH} \subseteq \mathbf{QMA}$, as desired. □

The consequence of this theorem can be strengthened from $\mathbf{PH} \subseteq \mathbf{QMA}$ to $\mathbf{PH}^{\mathbf{QMA}} \subseteq \mathbf{QMA}$.

Theorem 20. *If* $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ *then* $\mathbf{QMA} = \mathbf{PH}^{\mathbf{QMA}}$.

Proof. The proof is exactly identical to that of Theorem 18, with **QMA** in place of **PP** in the base and induction hypothesis. That is, the base case will be $\Sigma_1^{\mathbf{P}^{\mathbf{QMA}}} = \mathbf{QMA}$ and the induction step will assume that $\Sigma_i^{\mathbf{P}^{\mathbf{QMA}}} = \mathbf{QMA}$ and derive $\Sigma_{i+1}^{\mathbf{P}^{\mathbf{QMA}}} = \mathbf{QMA}$.

This gives $\mathbf{PH}^{\mathbf{QMA}} \subseteq \mathbf{QMA}$. For the opposite inclusion, simply note that $\mathbf{QMA} \subseteq \mathbf{P}^{\mathbf{QMA}} \subseteq \mathbf{PH}^{\mathbf{QMA}}$. □

Since we do not expect that **QMA** contains the entire polynomial hierarachy, much less the entire polynomial hierarchy with a **QMA** oracle, we take this as evidence that **QMA** is not closed under complement.

Note that for the proof of Theorem 19 it is not enough that the hypothesis implies that $\mathbf{NP} \subseteq \mathbf{co\text{-}QMA}$. The hypothesis that **QMA** is closed under complement is a crucial ingredient for the proof to go through. To illustrate,

Theorem 21. *If* $\mathbf{NP} \subseteq \mathbf{co\text{-}QMA}$ *then* $\Sigma_2^{\mathbf{P}} \subseteq \mathbf{QMA}$ *and more generally, then* $\Sigma_{i+1}^{\mathbf{P}} \subseteq \Sigma_i^{\mathbf{BQP}}$ *for every* i.

Proof. Suppose that $\mathbf{NP} \subseteq \mathbf{co\text{-}QMA}$. Then $\mathbf{NP} \subseteq \mathbf{QMA} \cap \mathbf{co\text{-}QMA}$, because we already knew $\mathbf{NP} \subseteq \mathbf{QMA}$. So $\mathbf{NP}^{\mathbf{NP}} \subseteq \mathbf{NP}^{\mathbf{QMA} \cap \mathbf{co\text{-}QMA}} \subseteq \mathbf{QMA}$ by Theorem 14. The general case, that $\Sigma_{i+1}^{\mathbf{P}} \subseteq \Sigma_i^{\mathbf{BQP}}$, is proven by induction. We

have just established the base case, namely $\Sigma_2^{\mathbf{P}} \subseteq \Sigma_1^{\mathbf{BQP}}$. In the induction step, we assume that $\Sigma_{i+1}^{\mathbf{P}} \subseteq \Sigma_i^{\mathbf{BQP}}$ and derive $\Sigma_{i+2}^{\mathbf{P}} \subseteq \Sigma_{i+1}^{\mathbf{BQP}}$.

$$\begin{aligned}
\Sigma_{i+1}^{\mathbf{P}} &\subseteq \Sigma_i^{\mathbf{BQP}} && \text{Induction hypothesis} && (2.66)\\
\Sigma_{i+2}^{\mathbf{P}} &= \mathbf{NP}^{\Sigma_{i+1}^{\mathbf{P}}} && \text{By definition} && (2.67)\\
&\subseteq \mathbf{NP}^{\Sigma_i^{\mathbf{BQP}}} && \text{Induction hypothesis} && (2.68)\\
&\subseteq \mathbf{QMA}^{\Sigma_i^{\mathbf{BQP}}} && \mathbf{NP} \subseteq \mathbf{QMA} \text{ relativizes} && (2.69)\\
&= \Sigma_{i+1}^{\mathbf{BQP}} && \text{By definition} && (2.70)
\end{aligned}$$

□

It is unclear how to proceed towards a proof that $\mathbf{PH} \subseteq \mathbf{QMA}$. There may be a way, but it will require a novel idea: in the proof as we have set it up, the hypothesis that $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ is necessary and sufficient.

2.4 Closure properties of QAM

The technique used above to establish the closure properties of **QMA** readily extends to showing closure properties of many other classes. In this section we prove results similar to Theorems 12 and 14 for the classes **AM** and **QAM**.

The class **AM** was introduced by Babai and Moran [11] as the class of languages decidable by a protocol, or rather a conversation, between a verifier, Arthur, and a prover, Merlin, as follows. First, Arthur asks Merlin a question, who responds with an answer. Arthur is allowed to use randomness to decide what question to ask. Arthur then performs a polynomial-time deterministic computation on the input, his random coins and Merlin's message. The protocol must yield the right answer with probability at least $\frac{2}{3}$, conditioned over the choice of Arthur's random coins. Without loss of generality, Arthur's message can be restricted to be simply his random seed, because Merlin can simulate the algorithm Arthur uses to produce his question.

Definition 3 AM: Arthur Merlin

A language $L \subseteq \{0,1\}^*$ is in **AM** if there is are polynomials $r(n)$ and $m(n)$ and a deterministic polynomial-time Turing Machine M such that, for all input strings $x \in \{0,1\}^*$:

- if $x \in L$, then there is a function $y\colon \{0,1\}^{r(n)} \to \{0,1\}^{m(n)}$ such that

$$\Pr_{s\in\{0,1\}^{r(n)}}[M(x,s,y(s)) = 1] \geq \frac{2}{3} \tag{2.71}$$

- If $x \notin L$ then for all functions $y(s)$,

$$\Pr_{s\in\{0,1\}^{r(n)}}[M(x,s,y(s)) = 0] \geq \frac{2}{3} \tag{2.72}$$

The class **QAM** was defined by Marriot and Watrous in [9] as the same class except Arthur now has a quantum computer at his disposal and Merlin may send him a quantum state. The protocol still begins with Arthur sending Merlin a polynomial number of classical random bits. Moreover, the probability of success is also conditioned on the inherent randomness in the quantum algorithm, in addition to Arthur's random coins sent to Merlin.

The literature also considers conversations with more than one round of questions, but we will not touch upon those here. The topic is known as "interactive proofs", as Merlin tries to prove a proposition to Arthur, and Merlin can respond interactively to Arthur's questions, as opposed to an oracle, which always answers questions the same way.

If the number of questions is bounded by a constant, then perhaps surprisingly, one obtains the class **AM** again. If the number of questions is unrestricted, one obtains the class **PSPACE** [32]. That is, if L is a language in **PSPACE** (for example, Chess), and Merlin claims that $x \in \{0,1\}^*$ (for example, he claims that white wins from board position x with perfect play), then whether that is true can be revealed with great confidence by a polynomial-time "interrogation" of Merlin. The quantum case behaves differently, if Arthur and Merlin are allowed to entangle their qubits, because then three rounds suffice for any purpose, i.e. **QMAM** = **PSPACE** [33, 54].

If there are multiple provers who do not communicate during the protocol but can agree on a strategy beforehand, but the protocol is still required to terminate in a polynomial amount of time, we obtain the class **NEXP** [41]. Quantum multi-

prover interactive proofs are known to be equally powerful [42]. However, it is an open problem to characterize the setting where the provers prepare an entangled state before the interrogation begins (that is, they share a number of EPR pairs), but otherwise do not communicate [43]. It is known only that in this setting, where the provers share EPR pairs, the quantum and classical is equally powerful.

These results all have the surprising implication that interaction greatly increases the computational powers of any verifier from **BPP** or **BQP** to **PSPACE** or **NEXP**, even if the verifier is skeptical of the prover(s) and demands that the probability of error is exponentially small.

Interactive proofs of great interest because they are one of the few known "non-relativizing techniques". As a result, they have been instrumental in proving circuit lower bounds that require defeating the relativization and natural proofs barriers, namely that **PP** does not have (quantum) circuits of size $O(n^k)$ [59, 25], that $\mathbf{MA_{EXP}}$ does not have polynomial-size circuits [61], and several circuit lower bounds against the promise version of **MA** [60]. Unfortunately, all techniques from interactive proofs fail to defeat the algebrization barrier [26], so better circuit lower bounds must develop new techniques.

In this Section, we prove that **QAM** is closed under Turing reductions, and we prove a statement analogous to Theorem 14. To warm up, we first show the proof for the classical case. Its structure is identical to the proof of Theorem 11.

Theorem 22. $\mathbf{P^{AM \cap co\text{-}AM}} = \mathbf{AM} \cap \mathbf{co\text{-}AM}$.

Proof. Let $L \in \mathbf{AM} \cap \mathbf{coAM}$ be a language and M^L be a deterministic $t(n)$-time oracle Turing Machine with oracle access to L. If we find an **AM** protocol to simulate M^L, we will have proved the theorem.

Since $L \in \mathbf{AM} \cap \mathbf{coAM}$, there are deterministic Turing Machines Y and N which execute the **AM** protocols for L and $\overline{L}$, respectively, using $r(n)$ random bits, running in time $t(n)$ and erring with probability $\leq e^{-n}$.

Again, of course, *if* we obtain the answers to all the queries M^L makes, *then* we can simulate M^L. We obtain those answers by running the machines Y and N, just as in the proof of Theorem 11, with two differences: (i) the protocol starts with Arthur generating some appropriate number of random bits and communicating those to Merlin, and (ii) the protocol may err with some small probability conditioned on those random bits.

The protocols starts with Arthur flipping $2t(n) \cdot r(t(n))$ coins and sending the result to Merlin. This number, $2t(n) \cdot r(t(n))$, is an upper bound on the number of random coins all the upcoming protocols need, as M^L makes at most $t(n)$ queries,

each at most $t(n)$ bits long. Then Y and N will use $r(t(n))$ random bits each to answer a query of length $t(n)$. We expect Merlin to send us (i) the answers $a_1, \ldots, a_{t(n)}$ to the queries M^L will make, (ii) the strings $q_1, \ldots, q_{t(n)}$ that M^L will query given the random coins we just guessed, and (iii) certificates $y_1, \ldots, y_{t(n)}$ and $z_1, \ldots, z_{t(n)}$ for the machines Y and N, respectively.

The first step for Arthur is to check whether the certificates are good. For each i, he checks that $Y(q_i, s_i, y_i) = a_i$ and $N(q_i, s_i, z_i) = \neg a_i$. Here s_i is the string of random coins Arthur flipped at the beginning to feed to the i-th query. If any of these checks fail, he rejects.

If all checks pass, then Arthur simulates M^L as before: If the i-th query of M^L is not the string q_i, he rejects; otherwise, he feeds M^L the answer a_i and resumes the simulation. When M^L halts, he copies its answer as his output.

Part 1: Completeness. Suppose that M^L accepts. If good certificates to our queries exist (a good certificate is one that Y will accept if the answer is *yes*), then Merlin will send them and Arthur will feed M^L correct answers and accept. However, the probability that such certificates exist, i.e. the probability that the **AM** protocol for q_i is successful, is only $1 - e^{-n}$, conditioned over the random bits Arthur generates at the beginning. The probability, then, that all **AM** protocols are successful, is $\geq (1 - e^{-n})^{2t(n)}$, which tends to 1 with $n \to \infty$ by Theorem 9 because $2t(n)$ is a polynomial.

Part 2: Soundness. Suppose that M^L rejects. If we feed M^L correct answers to its queries, we reject too. We only feed M^L incorrect answers if one of the **AM** protocols failed. Each **AM** protocol fails with probability $\leq e^{-n}$, and there are at most $2t(n)$ of them, so with overwhelming probability, all of them succeed. □

We now show that $\mathbf{QAM} \cap \mathbf{co\text{-}QAM}$ is closed under polynomial-time deterministic Turing reductions.

Theorem 23. $\mathbf{P}^{\mathbf{QAM} \cap \mathbf{co\text{-}QAM}} = \mathbf{QAM} \cap \mathbf{co\text{-}QAM}$

Proof. In this case there are two uniformly generated quantum circuit families $\{Y_s\}$ and $\{N_s\}$ which answer L and $\overline{L}$. We exchange random bits and certificates the same way we did in Theorem 22. We expect Merlin to send us (i) the answers to the queries, (ii) the strings that M^L is going to query and (iii) quantum certificates $|\psi\rangle = |\psi_1\rangle |\phi_1\rangle \otimes \cdots \otimes |\psi_{t(n)}\rangle |\phi_{t(n)}\rangle$ for the these quantum circuits. As in the proof of Theorem 12, we measure the bits that we expect to be classical bits before we execute the **QAM** protocols.

Part 1: Completeness. If M^L accepts, then Merlin will send us the correct answers and unentangled quantum certificates, if they exist, but as noted before,

good certificates exist with overwhelming probability. However, even if we give Y_s a good certificate, it may still err because it is a quantum circuit. Fortunately, it is known that **QAM** protocols such as Y_s can be amplified to err with $\leq e^{-n}$ error, so the previous completeness argument goes through.

Part 2: Soundness. By previous observations, for Arthur to fail it is necessary that at least one **QAM** protocol fails. In the classical case, the soundness of the protocols Y and N was sufficient to reduce the probability that any query failed. The difference in the quantum case is that Merlin can entangle the certificates. But if a quantum circuit Y_s rejects *all* input states with probability $\geq p$, then by the Entanglement Independence Lemma (Theorem 8) it rejects all states regardless of how they are entangled with other qubits that the circuit does not touch, with probability $\geq p$. In our case, $p \geq (1 - e^{-n})$, so we have soundness by the same argument as in Theorem 22. □

We now proceed to proving a theorem about **QAM** analogous to Theorem 14.

Theorem 24. $\mathbf{NP}^{\mathbf{AM} \cap \mathbf{co\text{-}AM}} \subseteq \mathbf{AM}$.

Proof. We will give an **AM** protocol for a language $A \in \mathbf{NP}^{\mathbf{AM} \cap \mathbf{co\text{-}AM}}$ accepted by nondeterministic Turing Machine M^L. We generate enough random bits, and Merlin responds with certificates to all the queries we are going to make, and with the nondeterministic bits which allegedly make M accept. Since the nondeterministic bits are fixed, the remaining computation is simply a $\mathbf{P}^{\mathbf{AM} \cap \mathbf{co\text{-}AM}}$ computation, which is covered in Theorem 22 □

Theorem 25. $\mathbf{NP}^{\mathbf{QAM} \cap \mathbf{co\text{-}QAM}} \subseteq \mathbf{QAM}$

Proof. Similar to the previous three theorems, a **QAM** simulation of a language in $\mathbf{NP}^{\mathbf{QAM} \cap \mathbf{co\text{-}QAM}}$ starts with sending enough random bits to Merlin and receiving a quantum state allegedly representing a classical certificate for the nondeterministic machine and quantum certificates so we can simulate the oracle queries. We measure these certificate bits in the computational basis, in addition to the qubits containing the queries and their alleged answers. The nondeterministic input is fixed now, so the remainder of the protocol simulates a deterministic machine making queries to a language in $L \in \mathbf{QAM} \cap \mathbf{co\text{-}QAM}$, which is covered in Theorem 23. □

Theorem 26. *If* $\mathbf{QAM} = \mathbf{co\text{-}QAM}$ *then* $\mathbf{PH} \subseteq \mathbf{QAM}$.

The proof is very similar to Theorem 19.

Proof. Clearly we have $\mathbf{NP} \subseteq \mathbf{QAM}$, and this inclusion relativizes. Therefore $\mathbf{NP}^{\mathbf{NP}} \subseteq \mathbf{NP}^{\mathbf{QAM} \cap \mathbf{co\text{-}QAM}} \subseteq \mathbf{QAM}$. The induction step is $\mathbf{NP}^{\Sigma_{i+1}^{\mathbf{P}}} \subseteq \mathbf{NP}^{\mathbf{QAM}} \subseteq \mathbf{QAM}$. Hence $\mathbf{PH} \subseteq \mathbf{QAM}$. □

2.5 Towards a quantum Toda's theorem

In the classical case, Toda's Theorem [10] says that $\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}$. Is this also true in the quantum case, i.e. is $\mathbf{BQPH} \subseteq \mathbf{P}^{\#\mathbf{P}}$? A version of this theorem is true, because it has already been proven that $\mathbf{BQP} \subseteq \mathbf{P}^{\#\mathbf{P}}$ and $\mathbf{QMA} \subseteq \mathbf{P}^{\#\mathbf{P}}$. (in fact, much better upper bounds are known for these classes: $\mathbf{BQP} \subseteq \mathbf{AWPP}$ and $\mathbf{QMA} \subseteq \mathbf{A0PP}$, but we concentrate on $\mathbf{P}^{\#\mathbf{P}}$ to see if the *entire* hierarchy is in there). No pun intended, but to take this to the next level, we ask: Is $\mathbf{BQP}^{\mathbf{QMA}} \subseteq \mathbf{P}^{\#\mathbf{P}}$? It is tempting to hope that simple relativizing inclusions and lowness properties suffice to prove this statement, but unfortunately, the bottleneck to that approach seems to be that we do not know (or expect) that **QMA** is low for **PP**. Therefore we must get our hands dirty and look into the proof of Toda's Theorem.

The step that looks hardest to adapt is the step where $\mathbf{NP} \subseteq \mathbf{RP}^{\oplus\mathbf{P}}$ is established by carefully counting the number of certificates an **NP** machine accepts. This number of accepting paths is a discrete, nonnegative integer. By constrast, the number of states that makes a quantum computer accept is uncountably infinite. Nevertheless, interesting progress in this direction has been made, of which we will now describe two recent developments.

First, the 1983 Valiant-Vazirani Theorem [13] has recently been extended to the quantum setting by Aharonov et. al [2]. Historically, it was a precursor to Toda's Theorem. It states that there is a random reduction from SAT to UniqueSAT, the problem of deciding whether a Boolean formula is satisfiable given the promise that it has at most one satisfying assignment. The recent extension is that there is a randomized reduction from the trivially **QCMA**-Complete problem of deciding whether there is a string that makes a **QCMA** machine accept, to the same problem but with the promise that there is at most one such string. Here **QCMA** is like **QMA** where we restrict the certificates to be classical strings. The principal difficulty in showing a similar result for the class **QMA** lies in establishing what kind of theorem we should even aim for, given that the solution space is continuous (One approach the authors suggest is to say that a quantum solution state is a unique solution if the circuit rejects all states orthogonal to it). This brings us to our next development.

Second, in [15], Bredariol et al. show that if the good certificates of a quantum

states are restricted to be uniform superpositions over the basis states, then the completeness property of **QMA** protocols is preserved. That is, when analyzing **QMA**, one may assume without loss of generality that every good certificate $|\psi\rangle_n$ of n qubits is related to some set $S \subseteq \{0,1\}^n$ in the following way (where $|S|$ denotes the size of S):

$$|\psi\rangle_n = \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle_n \tag{2.73}$$

This narrows the solution space down to only 2^{2^n} candidates, which is doubly-exponential in n, but at any rate finite (this is still not 2^n, as we may wish, but it is much better than $2^{\aleph_0}$!). It is interesting to see whether the two results above can be combined.

2.5.1 Towards collapsing the quantum polynomial hierarchy

We have tried and failed to show that our new hierarchy collapses if **QMA** = **co-QMA**. Classicaly this result follows almost trivially from the fact that $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ is low for **NP**, i.e. $\mathbf{NP}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} = \mathbf{NP}$. The classical proof is as follows.

Theorem 27. *If* $\mathbf{NP} = \mathbf{co\text{-}NP}$ *then* $\mathbf{PH} = \mathbf{NP}$.

Proof. Suppose that **NP** is closed under complement

$$\begin{aligned}
\mathbf{NP} &= \mathbf{co\text{-}NP} && \text{The assumption} && (2.74)\\
\mathbf{NP} &= \mathbf{NP} \cap \mathbf{co\text{-}NP} && \text{Implied by assumption} && (2.75)\\
\Sigma_2^P &= \mathbf{NP}^{\mathbf{NP}} && \text{By definition} && (2.76)\\
&= \mathbf{NP}^{\mathbf{NP} \cap \mathbf{co\text{-}NP}} && \text{By line 2.75} && (2.77)\\
&= \mathbf{NP} && \text{Because } \mathbf{NP} \cap \mathbf{co\text{-}NP} \text{ is low for } \mathbf{NP} && (2.78)
\end{aligned}$$

And there we go! Easy as pie! The higher levels of the polynomial hierarchy collapse by induction, like so:

$$\begin{aligned}
\Sigma_3^P &= \mathbf{NP}^{\Sigma_2^P} && \text{By definition} && (2.79)\\
&= \mathbf{NP}^{\mathbf{NP}} && \text{We just proved } \Sigma_2^P = \mathbf{NP} && (2.80)\\
&= \Sigma_2^P = \mathbf{NP} && && (2.81)
\end{aligned}$$

□

In this subsection, we show where exactly the classical proof breaks down. It will suffice to analyze why $\mathbf{P}^{\mathbf{NP}\cap\mathbf{co\text{-}NP}} \subseteq \mathbf{NP}$ is easy to prove but $\mathbf{BPP}^{\mathbf{NP}\cap\mathbf{co\text{-}NP}} \subseteq \mathbf{NP}$ is not.

If the simulated machine is deterministic, then our nondeterministic simulator simply runs the machine and only needs its non-determinism to handle the machine's oracle queries. However, if the simulated machine is randomized, then the different branches may query different strings. Which queries should the nondeterministic machine answer? Which brances should it even investigate? For example, the **BPP** machine may flip a coin and ask the oracle for a Hamiltonian cycle in a graph which depends on the coin. Then even if both graphs contain a Hamiltonian cycle, there is not necessarily *one* message that would satisfy both queries (of course, the prover could send both cycles. But that strategy breaks down when the simulated machine generates a superpolynomial number of computation paths). At the time of writing, derandomization has not yet shown that $\mathbf{BPP} \subseteq \mathbf{NP}$, so there is no generic nondeterministic simulation of randomized algorithms.

The same difficulty presents itself in the quantum case. Where a classical randomized machine flips a coin and obtains a bit $a \in \{0,1\}$, a quantum machine may put a qubit in the superposition $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and ask a query which depends on that qubit. Then Merlin's dilemma is the same: even if the answers to both queries are *yes*, it is not obvious what certificate state would convince Arthur.[2]

A natural next step is to try to prove that the quantum polynomial hierarchy collapses under a stronger hypothesis than $\mathbf{QMA} = \mathbf{co\text{-}QMA}$. For example, does $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ and $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ suffice to collapse the hierarchy? Not as far as we know. We do not even know how to prove $\mathbf{BQP} = \mathbf{QCMA}$ under this hypothesis. We know how to prove something almost as good, however, namely if $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ then $\mathbf{BQP} = \exists \cdot \mathbf{BQP}$. Here $\exists \cdot \mathbf{BQP}$ is the class **BQP**with a classical existential quantifier. It consists of all languages L for which there is a language $K \in \mathbf{BQP}$ and a polynomial function $p(n)$ such that

$$x \in L \iff \exists y \in \{0,1\}^{p(n)} \colon \langle x, y\rangle \in K \tag{2.82}$$

It is easy to prove from the definition of the classes that $\mathbf{NP} = \exists \cdot \mathbf{P}$. It may look obvious that $\exists \cdot \mathbf{BPP} = \mathbf{MA}$, but in fact there is an oracle due to Fenner,

[2]If Merlin is able to entangle his certificate state with Arthur's query bits, then he can convince Arthur by constructing the state $|0\rangle|\psi_0\rangle + |1\rangle|\psi_1\rangle$, where $|\psi_i\rangle$ is the certificate state for the query $|i\rangle$. Such interactive proof systems are studied in the literature[9], but they are too powerful for our purposes.

Fortnow and Kurtz [52], relative to which $\exists \cdot \mathbf{BPP} \subsetneq \mathbf{MA}$. The catch is that a Turing Machine in an **MA** protocol is allowed to accept non-certificates for yes-instances with arbitrary probability, such as $1/2$ or 0, so long as there exists a good certificate it accepts with high probability. In $\exists \cdot \mathbf{BPP}$, however, the machine must always either accept or reject with high probability. For the same reason, it is unknown whether $\exists \cdot \mathbf{BQP} = \mathbf{QCMA}$.

Theorem 28. *If* $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ *then* $\mathbf{BQP} = \exists \cdot \mathbf{BQP}$.

Proof. The $\exists$ operator operates purely syntactically on languages and sets of languages, and in fact has no relation to the underlying models of computation used to define those languages. Under the assumption, $\mathbf{P} = \mathbf{BQP}$. Since $\mathbf{NP} = \exists \cdot (\mathbf{P})$, we have $\exists \cdot (\mathbf{BQP}) = \exists \cdot (\mathbf{P}) = \mathbf{NP} = \mathbf{P}$. □

Clearly $\exists \cdot \mathbf{BQP} \subseteq \mathbf{QCMA} \subseteq \mathbf{QMA}$. The next step would be to prove that $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ implies $\mathbf{P} = \mathbf{QCMA}$. The principal difficulty lies in how to handle certificates that are accepted with probabilities in $[1/3, 2/3]$. In [2], Aharonov et al. overcome precisely this obstacle to establish a quantum Valiant-Vazirani theorem. We have good hope that their technique can be useful here as well.

2.6 Related work and open problems

The guiding motivation in studying the quantum hierarchy in this thesis is to supply tools that can be of help elsewhere in complexity. So the biggest open question is: What questions can this object help answer? For example, it is an open question to find an oracle which puts **BQP** outside the polynomial hierarchy, i.e. $\mathbf{BQP} \neq \mathbf{PH}$. This chapter suggests a less ambitious goal: put any level of the quantum polynomial hierarchy outside the classical polynomial hierarchy, e.g. $\Delta_3^{\mathbf{BQP}} \neq \mathbf{PH}$.

Conjecture 1. *There is an oracle relative to which* $\mathbf{PH} \neq \mathbf{BQPH}$.

It is an open question to find an implication either way between $\mathbf{P} = \mathbf{NP}$ and $\mathbf{BQP} = \mathbf{QMA}$. We would like to suggest first finding an implication conditioned upon $\mathbf{P} = \mathbf{BQP}$. For example, in this chapter we have pointed out the obvious truth that if $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ then $\mathbf{P} = \exists \cdot \mathbf{BQP}$. A natural question is whether that hypothesis implies $\mathbf{P} = \mathbf{QCMA}$, and from there we would like to see whether (or not!) it implies $\mathbf{P} = \mathbf{QMA}$. The principal difficulty in adapting the proof is

that a **QCMA** machine is allowed to accept a bad certificate for a yes-instance with arbitrary probability.

Conjecture 2. *If* $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ *then* $\mathbf{P} = \mathbf{QCMA}$.

The difficulty in adapting a proof for the conjecture above, if indeed it is true, to the case $\mathbf{P} = \mathbf{QMA}$ will lie in the fact that Merlin can choose to send quantum states that no polynomial-sized quantum circuit can prepare with the necessary accuracy. In fact, for this reason, we conjecture that the implication does *not* hold:

Conjecture 3. *There is an oracle relative to which* $\mathbf{P} = \mathbf{NP} = \mathbf{BQP}$ *and yet* $\mathbf{P} \subsetneq \mathbf{QMA}$.

(Of course, in the unrelativized world, the conjecture is likely to hold by virtue of its hypothesis being false!).

We have failed to prove whether $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ implies that the quantum polynomial hierarchy collapses. Is this true? We have seen that this hypothesis implies $\mathbf{PH} \subseteq \mathbf{QMA}$; that corollary would be a triviality if the conjecture were true, so we take it as evidence in favour of the conjecture.

Conjecture 4. *If* $\mathbf{QMA} = \mathbf{co\text{-}QMA}$ *then the quantum polynomial hierarchy collapses.*

Toda's Theorem [10] states that the polynomial hierarchy is not harder than counting solutions to Boolean formulae, i.e. $\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}} = \mathbf{P}^{\mathbf{PP}}$. We have not found a way to quantumize Toda's theorem. We expect, however, that it is possible to do so, so we conjecture the following:

Conjecture 5. $\mathbf{BQP}^{\mathbf{QMA}} \subseteq \mathbf{P}^{\mathbf{PP}}$ *and indeed* $\mathbf{BQPH} \subseteq \mathbf{P}^{\mathbf{PP}}$.

Interesting progress in this direction has been made by Aharonov et. al in [2], who provide a quantum analogue of the Valiant-Vazirani Theorem, which historically was the first step towards Toda's theorem. We see extending their result as the logical next step towards establishing a quantum version of Toda's theorem.

One of the first non-trivial unconditional circuit lower bounds was that of Kannan [20], which states that for each $k \in \mathbb{N}$, the class Σ_2^P does not have (classical) circuits of size $O(n^k)$ for any k (that is, there is a language $L \in \Sigma_2^P$ such that no circuit family whose size is bounded by n^k solves L). Can we prove the natural quantum analogue?

Conjecture 6. *The class* $\mathbf{QMA}^{\mathbf{QMA}}$ *does not have quantum circuits of size* n^k *for any* k*.*

Lastly, one conditional collapse seems to have already been partially established, namely a quantum version of the Karp-Lipton collapse. Recall the Karp-Lipton Theorem:

Theorem 29 (Karp-Lipton Theorem [21])**.** *If* $\mathbf{NP} \subset \mathbf{P}_{/poly}$ *then* $\Sigma_2^P = \Pi_2^P$*.*

In [7], Aaronson and Drucker establish a quantum analogue:

Theorem 30. *Quantum Karp-Lipton Theorem [7] If* $\mathbf{NP} \subset \mathbf{BQP}_{/qpoly}$ *then* $\Pi_2^P \subseteq \mathbf{QMA}^{\mathbf{PromiseQMA}}$*.*

The authors state that it was the first nontrivial result which shows that quantum computers being able to solve **NP**-Complete problems in polynomial time has unlikely consequences (In discussion with Aaronson, his opinion is that if the quantum polynomial hierarchy is to be a useful tool in complexity theory, then this is the kind of result that would make that case). It is natural to ask whether and how his theorem generalizes to higher levels of the quantum polynomial hierarchy.

Chapter 3

A QMA-Complete problem: the Local Hamiltonian

The class **BQP** formalizes the notion of an efficient algorithm, just as **P** formalizes the notion of an efficient classical algorithm. Just as **NP** is an attempt to capture efficient classical verifiability, the quantum analogue **QMA** attempts to capture the same for quantum computers. Indeed the $\mathbf{BQP} \stackrel{?}{=} \mathbf{QMA}$ question is typically regarded as the quantum version of the $\mathbf{P} \stackrel{?}{=} \mathbf{NP}$ question.

The class **QMA** was first defined by Kitaev [62], and he showed that there is a quantum generalization of the celebrated Cook-Levin theorem, which establishes that the class **NP** has complete problems, by exhibiting a **QMA**-Complete problem, the Local Hamiltonian Problem. This problem is a natural extension of constraint satisfaction problems, placing constraints on qubits rather than bits. This class is of theoretical interest, but also of practical importance, because it captures problems related to quantum physics that arise in practice but cannot be characterized by **NP** or indeed by the polynomial hierarchy, as those objects deal only with classical computation. For example, the Local Hamiltonian Problem determines the ground state energy of a quantum state. This chapter proves that the Local Hamiltonian Problem is **QMA**-Complete.

Many **QMA**-Complete problems have been described since. Most of them are natural analogues of well-known **NP**-Complete problems. For example, determining whether a quantum circuit implements the identity is **QMA** complete, whereas determining whether a classical circuit (with n inputs and n outputs) implements the identity is **NP**-Complete. It also is **QMA**-Complete to decide whether two quantum circuits implement the same unitary operation, whereas it is **NP**-Complete to decide whether two classical circuits implement the same

Boolean function.

The Local Hamiltonian Problem has a parameter called its *locality*, which is the maximum number of terms (qubits) which may occur together in a single constraint. Kitaev's original proof showed that 5-Local Hamiltonian was **QMA**-Complete. Subsequently, in 2003, Kempe and Regev reduced the locality to $k = 3$. Finally, in 2006, Kitaev, Kempe and Regev showed that even 2-Local Hamiltonian is **QMA**-Complete. It is easy to show that 1-Local Hamiltonian is solvable classically in polynomial time, so it is generally believed that, as far as **QMA**-Completeness is concerned, $k = 2$ is rock-bottom. Later still, in 2016, Cubitt and Montanaro provided a classification of all local Hamiltonian problems, much like Schaeffer's Dichotomy Theorem does for variations of SAT [45].

All these results together go some way in answering the question *"Does the quantum polynomial hierarchy look like the classical polynomial hierarchy*, by ascertaining that there is a quantum analogue to the class of **NP**-Complete problems.

Before we tackle the issue of whether the Local Hamiltonian Problem is in **QMA**, we warm up with a few preliminaries. First, we review how projective operator-valued measurements work and review a lemma about Hermitian matrices. Next, we examine a quantum circuit which performs a fair m-sided coin toss.

Our proof is a detailed exhibition of proofs by Kitaev [62] and by Aharonov and Naveh [63], tailored to be accessible to readers with very little background in linear algebra.

3.0.1 Preliminaries on Projection operators

A projection operator is a way to take a vector and preserve only its amplitude in certain directions and filter out its amplitude in certain other directions. For example, the operator $|0\rangle_1 \langle 0|_1$, operating on $|\phi\rangle_1 = \alpha |0\rangle_1 + \beta |1\rangle_1$, yields

$$|0\rangle_1 \langle 0|_1 (\alpha |0\rangle_1 + \beta |1\rangle_1) = \alpha |0\rangle_1 \langle 0|_1 |0\rangle_1 + \beta |0\rangle_1 \langle 0|_1 |1\rangle_1 = \alpha |0\rangle_1$$

This way, only the component of $|\phi\rangle_1$ in the direction $|0\rangle_1$ is preserved. We say that *"$|\phi\rangle_1$ is projected onto the subspace spanned by $|0\rangle_1$"*. It is possible to project vectors onto multi-dimensional subspaces. Consider the vector

$$|\zeta\rangle_2 = \alpha |00\rangle_2 + \beta |01\rangle_2 + \gamma |10\rangle_2 + \delta |11\rangle_2$$

Suppose we wished to retain only the components in the directions $|00\rangle_2$ and $|10\rangle_2$, in other words to project $|\zeta\rangle_2$ onto the subspace spanned by $|00\rangle_2$ and $|10\rangle_2$. Clearly the result should be the vector $\alpha |00\rangle_2 + \gamma |10\rangle_2$. Therefore the appropriate projection operator is $|00\rangle_2 \langle 00|_2 + |10\rangle_2 \langle 10|_2$. This projection can be thought of as preserving the part of the system that is consistent with the first qubit having a value of $|0\rangle_1$.

In general, to project an n-qubit system onto the subspace in which the qubits $i, \ldots, i+k-1$ are in state $|\phi\rangle_k$, we will use the following projection operator:

$$\Pi_{n,i}^{|\phi\rangle_k} = I^{\otimes i} \otimes |\phi\rangle_k \langle\phi|_k \otimes I^{\otimes n-k-i}$$

Consider the state $|\zeta\rangle_2$ again. It is equal to the sum of (i) its component consistent with the first qubit being in state $|0\rangle_1$ and (ii) its component consistent with the first qubit being in state $|1\rangle_1$. This is a more general phenomenon: if $|\phi_0\rangle_k, \ldots, |\phi_{2^k-1}\rangle_k$ is an orthonormal base for $\mathcal{B}^{\otimes k}$, then, for any state $|\psi\rangle_n$,

$$|\psi\rangle_n = \Pi_{n,i}^{|\phi_0\rangle_k} |\psi\rangle_n + \cdots + \Pi_{n,i}^{|\phi_{2^k-1}\rangle_k} |\psi\rangle_n = \left(\sum_{u=0}^{2^k-1} \Pi_{n,i}^{|\phi_u\rangle_k} \right) |\psi\rangle_n$$

Hence, summing all these operators gives the identity operator:

$$\sum_{u=0}^{2^k-1} \Pi_{n,i}^{|\phi_u\rangle_k} = I^{\otimes i} \otimes \sum_{u=0}^{2^k-1} |\phi_u\rangle_k \langle\phi_u|_k \otimes I^{\otimes n-k-i} = I^{\otimes n}$$

Projection operators are useful when describing quantum measurements. Given a quantum state $|\psi\rangle_n$, the probability of obtaining a state $|\phi\rangle_k$ when measuring qubits $i, \ldots, i+k-1$ is the squared amplitude of the component of $|\psi\rangle_n$ consistent with that result, in other words the component of $|\psi\rangle_n$ in the direction $|\phi\rangle_k$:

$$P\left[|\phi\rangle_k\right] = \left| \Pi_{n,i}^{|\phi\rangle_k} |\psi\rangle_n \right|^2$$

With a little more work, one can obtain expressions for scenarios in which the qubits that are measured are not adjacent, but we will not need it here. We will need the following elementary theorems from linear algebra.

Theorem 31. *Every Hermitian n-dimensional operator H has an n-dimensional eigenspace, and can be expressed as*

$$H = \sum_{i=0}^{n-1} \lambda_i \left|\psi_i\right\rangle_n \left\langle\psi_i\right|_n = \sum_{i=0}^{n-1} \lambda_i \Pi_{n,0}^{\left|\psi_i\right\rangle_n},$$

where $\left|\psi_i\right\rangle_n$ *is an eigenvector with eigenvalue* λ_i*.*

Theorem 32. *The eigenvalues of a Hermitian operator are all real.*

Theorem 33. *If* $H = \sum_i \lambda_i \left|\psi_i\right\rangle \left\langle\psi_i\right|$ *is a Hermitian operator whose eigenvalues* λ_i *are all bigger than* τ*, then* $\left\langle\mu\right| H \left|\mu\right\rangle > \tau$ *for all unit vectors* $\left|\mu\right\rangle$*.*

Theorem 34. *Projections only have the eigenvalues* 1 *and* 0*.*

Theorem 35. *if* A *and* B *are non-negative operators, meaning* $\left\langle v\right| A \left|v\right\rangle$ *and* $\left\langle v\right| B \left|v\right\rangle$ *for all vectors* $\left|v\right\rangle$*, then* $C = A + B$ *is also a non-negative operator.*

Proof. Let $\left|v\right\rangle$ be any vector. Then $\left\langle v\right| C \left|v\right\rangle = \left\langle v\right| (A+B) \left|v\right\rangle = \left\langle v\right| A \left|v\right\rangle + \left\langle v\right| B \left|v\right\rangle$ is the sum of two non-negative real values, so $\left\langle v\right| C \left|v\right\rangle \geq 0$ is also non-negative. □

Theorem 36. *If* Π_1, Π_2 *are projections with* $\Pi_1\Pi_2 = 0$*, then the operator* $A = \Pi_1 + \Pi_2$ *only has the eigenvalues* 0 *and* 1*.*

3.0.2 Selecting a random qubit

The Hamming Weight $w(s)$ of a string s is the number of ones in that string. We will endeavour to find a circuit which takes n qubits $\left|s\right\rangle_n$ in the standard basis and outputs a qubit such that the probability of measuring this qubit in state $\left|1\right\rangle$ is $\frac{w(s)}{n}$. That is, the transformation acts as follows on the vectors $\left|s\right\rangle_n$ of the standard basis and one workspace qubit initialised to $\left|0\right\rangle_1$: [1]

$$\left|s\right\rangle_n \left|0\right\rangle_1 \to \left|s\right\rangle_n \otimes \left(\sqrt{1 - \frac{w(s)}{n}} \left|0\right\rangle_1 + \sqrt{\frac{w(s)}{n}} \left|1\right\rangle_1\right) \tag{3.1}$$

We leave it as an excercise to verify that (i) the probability of measuring the last qubit in state $\left|1\right\rangle$ after application of this circuit and (ii) the probability of measuring the k-th qubit in state $\left|1\right\rangle$, having chosen k by tossing a fair n-sided die, are the same probability, even when the input state is not one of the standard basis vectors.

[1] We will follow our convention of specifying the action of the operator only on states whose workspace is properly initialised. For example, we do not care how the operator above acts on the state $\left|s\right\rangle_n \left|1\right\rangle_1$; filling in any value will do (so long as the operator remains unitary) and there are many ways to do so.

3.1 The Local Hamiltonian Problem

Definition 4 k-local Hamiltonian Problem

Input: A series of T Hermitian operators $H_t \colon \mathcal{B}^{\otimes n} \to \mathcal{B}^{\otimes n}$, each of which operates nontrivially on at most k qubits, and two numbers a and b, with $0 \leq a < b \leq 1$.
Output: *"Yes"* if the Hamiltonian $\mathcal{H} = \sum_t H_t$ has an eigenvalue of at most a, or *"No"* if H only has eigenvalues larger than b, with the promise that one of these conditions holds.

The lowest eigenvalue of a Hamiltonian is called its *ground state energy* and the corresponding eigenvector(s) is called the *ground state*. To get an intuition for the Local Hamiltonian problem, let's first see why it is **NP**-Hard.

Theorem 37. *The* 3-Local Hamiltonian *problem is* **NP***-Hard*

Proof. We give a reduction from the **NP**-Complete problem 3-SAT: we are given a Boolean formula ϕ in conjunctive normal form, and our job is to output a Local Hamiltonian $\mathcal{H}$ and numbers a and b such that if ϕ is satisfiable, then H has an eigenvalue of a or less, whereas if ϕ is not satisfiable, then all eigenvalues of H are greater than or equal to b.

The Hamiltonian will work on n qubits: one for each variable in ϕ. We will take each clause and transform it into a Hermitian term, and $\mathcal{H}$ will simply be the sum of these terms. The idea will be that each term "penalizes" states that violate a clause by increasing the value of $\|H|\psi\rangle\|$. If $|\psi\rangle$ is a superposition, then the terms will penalize those branches of the wavefunction that violate the clause. We now make this formal.

If the t-th clause is

$$c_t = (x_3 \vee \neg x_5 \vee \neg x_{11}) \tag{3.2}$$

then we set the t-th local Hermitian operator to the $2^3 \times 2^3$ matrix

$$H_t = (|0\rangle_1 |1\rangle_1 |1\rangle_1) \cdot (\langle 0|_1 \langle 1|_1 \langle 1|_1) \qquad \text{On qubits } 3, 5, 11 \tag{3.3}$$

That is, the Hermitian operator is the outer product of $|011\rangle_3$. The action of H_t on the basis vectors is

$$H_t |x\rangle_n = \begin{cases} |x\rangle_n & \text{If } x \text{ violates clause } c_t \\ \vec{0} & \text{If } x \text{ satisfies clause } c_t \end{cases} \tag{3.4}$$

The Hamiltonian H is simply the sum of the Hermitians corresponding to each clause, and the threshold numbers are set to $a = 0$ and $b = 1$. The individual Hermitian operators act non-trivially only on exactly three qubits at a time, so they are 3-local, as intended.

$$\mathcal{H} = \sum_{t=1}^{m} H_t \tag{3.5}$$

Part 1: Completeness. Suppose that ϕ is satisfiable. Then there is some satisfying assignment $x \in \{0,1\}^n$. Then the lowest eigenvalue of H is obtained at $|x\rangle_n$:

$$\mathcal{H} |x\rangle_n = \sum_{t=1}^{m} H_t |x\rangle_n = \sum_{t=1}^{m} \vec{0} = \vec{0} \tag{3.6}$$

Indeed, the number $\langle x| H |x\rangle$ is precisely the number of violated clauses, so for any satisfying assignment x this number will be 0. The eigenvectors of H with eigenvalue 0 are precisely the satisfying assignments to ϕ. [2] Because every satisfiable formula is mapped to a Hamiltonian with eigenvalue 0, we conclude that our construction has completeness.

Part 2: Soundness. Suppose that ϕ is not satisfiable. Then for every bitstring $x \in \{0,1\}^n$, there is some clause that it does not satisfy. Let $v\colon \{0,1\}^n \to [m]$ be a function which, for each assignment, outputs the index of one of the violated clauses. Then the action of H on the basis vectors is as follows:

$$\langle x|_n H |x\rangle_n = \sum_{t=1}^{m} \langle x|_n H_t |x\rangle_n \geq \langle x|_n H_{v(x)} |x\rangle_n = 1 \tag{3.7}$$

We can use some elementary linear algebra to conclude that H has a trivial kernel, and therefore 0 is not one of its eigenvalues. Nevertheless, let's do the computation ourselves for general quantum states. Suppose Merlin gave an alleged certificate $|\psi\rangle_n = \sum_{x=0}^{2^n-1} y_x |x\rangle_n$, some highly entangled superposition, and claims that $|\psi\rangle_n$ is an eigenvector of H with a low eigenvalue. Denote with $k(x)$ the number of

[2] More precisely, the satisfying assignments to ϕ span the kernel of H, so states that are superpositions of satisfying assignments are also zero eigenvectors of H.

clauses violated by the assignment x. Then

$$\langle\psi|_n H |\psi\rangle_n = \left(\sum_{x=0}^{2^n-1} y_x^\dagger \langle x|_n\right) \cdot \left(\sum_{t=1}^{m} H_t\right) \cdot \left(\sum_{z=0}^{2^n-1} y_z |z\rangle_n\right) \tag{3.8}$$

$$= \sum_{x=0}^{2^n-1}\sum_{z=0}^{2^n-1} y_x^\dagger y_z \sum_{t=1}^{m} \langle x|_n H_t |z\rangle_n \tag{3.9}$$

$$= \sum_{x=0}^{2^n-1}\sum_{z=0}^{2^n-1} y_x^\dagger y_z \langle x|_n \cdot \sum_{t=1}^{m} \begin{cases} |z\rangle_n & \text{z violates } c_t \\ \vec{0} & \text{z satisfies } c_t \end{cases} \tag{3.10}$$

$$= \sum_{x=0}^{2^n-1}\sum_{z=0}^{2^n-1} y_x y_z k(z) \langle x|z\rangle \tag{3.11}$$

$$= \sum_{x=0}^{2^n-1} |y_x|^2 \cdot k(x) \tag{3.12}$$

$$\geq \sum_{x=0}^{2^n-1} |y_x|^2 \cdot 1 = 1 \tag{3.13}$$

So indeed any vector $|\psi\rangle$ has $\langle\psi| H |\psi\rangle \geq 1$, which completes the proof. □

The basic idea in the proof above is that $\langle\psi| H |\psi\rangle$ is a "penalty function" which is to be minimized, and that the terms of H aim to penalize wrong states. This will also be the idea when we construct the Hamiltonian in the proof that the Local Hamiltonian Problem is **QMA**-Hard. Before we continue, let's show the slightly stronger result that 2-Local Hamiltonian is also **NP**-Hard, by reducing from the **NP**-Complete problem Max2-SAT. Recall that Max2SAT is the problem to determine whether a given 2SAT formula has an assignment satisfying k or more clauses.

Theorem 38. *The problem* 2-Local Hamiltonian *is* **NP***-Hard.*

Proof. Given a Max2SAT instance (ϕ, k), we use the construction above for H. If x is an assignment satisfying at least k clauses, then it violates at most $m-k$ clauses, so $\langle x| H |x\rangle \leq m-k$. If no such assignment exists, and all assignments satisfy less than k clauses, then they violate at least $m-k+1$ clauses, so all states have $\langle\psi| H |\psi\rangle \geq m-k+1$.

According to our definition, we must have $a, b \leq 1$, and the values for $\langle\psi| H |\psi\rangle$ are all greater than 1, so we cannot simply set $a = m-k$ and $b = m-k+1$. We can repair this by normalizing our Hamiltonian with $\frac{1}{m}$, so our output is $(\frac{1}{m}H, \frac{m-k}{m}, \frac{m-k-1}{m})$. □

We now prove that there is a Merlin-Arthur protocol for the k-local Hamiltonian Problem with a small error rate. The theorem is due to Kitaev [62], and later received a beautiful exposition by Aharonov and Naveh [63]. We will follow these constructions closely.

Theorem 39. *The k-local Hamiltonian Problem can be solved with small probability of error, in time that is exponential in k, polynomial in the number T of Hermitian operators and polynomial in $\frac{1}{b-a}$, given a message of untrusted advice in the form of a quantum state from Merlin. That is to say, it is in* **PromiseQMA**.

Proof. The central insight is that it is possible to build a quantum circuit which takes a Hermitian operator H and a quantum state $|\mu\rangle_n$ and rejects with probability $\langle\mu| H |\mu\rangle$. In this sense it may be said that the circuit *evaluates* $\langle\mu| \mathcal{H} |\mu\rangle$. Hence if we choose to evaluate one of the T Hermitian operators $H_1, \ldots, H_T$ at random, then we will reject with probability $\frac{1}{T} \langle\mu| H |\mu\rangle$:

$$P(0) = \frac{1}{T} \sum_{t=1}^{T} \langle\mu| H_t |\mu\rangle = \frac{1}{T} \langle\mu| \left(\sum_{t=1}^{T} H_t \right) |\mu\rangle = \frac{1}{T} \langle\mu| H |\mu\rangle \tag{3.14}$$

The classical analogue of this algorithm would be to receive a formula on m clauses and an assignment $|x\rangle$, to choose one of the clauses at random, and reject if the assignment violates that clause, which happens with probability $\frac{k(x)}{m} = \frac{1}{m} \langle x| H |x\rangle$.

For the sake of clarity, let us first demonstrate the simplest case in which we are given one Hermitian operator $\mathcal{H}$, which acts on all the n qubits and has an eigenspace decomposition of $\mathcal{H} = \sum_{s=0}^{2^n-1} \lambda_s |\psi_s\rangle_n \langle\psi_s|_n$. We have prepared or received a state $|\mu\rangle_n = \sum_{s=0}^{2^n-1} \alpha_s |\psi_s\rangle_n$, and we wish to evaluate $\langle\mu|_n \mathcal{H} |\mu\rangle_n$, i.e. we wish to build a circuit which rejects with probability $\langle\mu| H |\mu\rangle$. Then we prepare an ancilla qubit in the state $|0\rangle_1$ and apply the operation U, which acts as follows on the basis of eigenvectors of $\mathcal{H}$:

$$U |\psi_s\rangle_n |0\rangle_1 = |\psi_s\rangle_n \otimes \left(\sqrt{\lambda_s} |0\rangle_1 + \sqrt{1-\lambda_s} |1\rangle_1 \right) \tag{3.15}$$

Then the probability of measuring the ancilla qubit in state $|0\rangle_1$, i.e. the probability that the machine rejects, is

$$P(|0\rangle) = \left| \Pi_0^{|0\rangle} U |\mu\rangle_n |0\rangle_1 \right|^2 = \left| \sum_{s=0}^{2^n-1} \alpha_s \sqrt{\lambda_s} |\Psi_s\rangle_n |0\rangle_1 \right|^2 = \sum_{s=0}^{2^n-1} |\alpha_s|^2 \lambda_s$$

To verify that this operator U achieves our goal, we now write out $\langle\mu|_n \mathcal{H} |\mu\rangle_n$,

$$\langle\mu| \mathcal{H} |\mu\rangle_n = \sum_{u=0}^{2^n-1} \langle\psi_u|_n \alpha_u^\dagger \sum_{s=0}^{2^n-1} \lambda_s |\psi_s\rangle_n \langle\psi_s|_n \sum_{a=0}^{2^n-1} \alpha_a |\psi_a\rangle_n = \sum_{s=0}^{2^n-1} |\alpha_s|^2 \lambda_s \quad (3.16)$$

We see that indeed, this circuit rejects with probability $P(|0\rangle) = \langle\mu|_n \mathcal{H} |\mu\rangle_n$.

Let us now turn to the general case. We are given T Hermitian operators $H_1, \ldots, H_T \colon \mathbb{B}^{\otimes n} \to \mathbb{B}^{\otimes n}$ each acting on a space of n qubits, but acting nontrivially on only k qubits. Each of these Hermitians H_t is given to us as a $2^k \times 2^k$ matrix M_t and an index s_t, so that this operator acts on $s_t, s_t + 1, \ldots, s_t + k - 1$. The Hermitians H_t on n qubits and M_t on k qubits are related by

$$H_t = I^{\otimes s_t} \otimes M_t \otimes I^{\otimes n-k-s_t}$$

First, for each Hermitian operator M_t, find an orthonormal basis of 2^k eigenvectors $\psi_{t,0}, \ldots, \psi_{t,2^k-1}$ with corresponding eigenvalues $\lambda_{t,0}, \ldots, \lambda_{t,2^k-1}$. We know from the lemma above that such a base always exists, because M_t is Hermitian. Finding such a base may take time exponential in k, but since k is a problem-specific constant, this is not a problem. We will return to this issue later, because it is an important detail when other problems are reduced to the k-local Hamiltonian.

We will assume that all eigenvalues are nonnegative, and at most 1. If we find a negative eigenvalue, we simply reject. We will show later that this is not an obstacle later, when we show that the Local Hamiltonian Problem is **QMA**-Complete.

Using the orthonormal base for M_t, construct a quantum circuit W_t on $k + 1$ qubits. The circuit W_t acts nontrivially on the same qubits as the Hermitian operator H_t, and one additional output qubit. On the qubits $s_t, \ldots, s_t + k - 1$, on which H_t does not act trivially, the action of W_t is described in terms of the base vectors $|\psi_{t,u}\rangle_k$, as follows:

$$|x\rangle_{s_t} |\psi_{t,u}\rangle_k |y\rangle_{n-k-s_t} |0\rangle_1 \overset{W_t}{\mapsto} |x\rangle_{s_t} |\psi_{t,u}\rangle_k |y\rangle_{n-k-s_t} \otimes \left(\sqrt{\lambda_{u,t}} |0\rangle_1 + \sqrt{1 - \lambda_{u,t}} |1\rangle_1\right) \quad (3.17)$$

Here $0 \leq u \leq 2^k - 1$. (Again, the size of this circuit may well be exponential in k, but since k is a constant, the size of this circuit is bounded by a constant, so this need not bother us.)

Consider the circuit W_t in isolation, without the other circuits. If we use only this circuit W_t, then Merlin can send us the state $|\mu\rangle_n = |0\rangle_{s_t} |\psi_{t,u}\rangle_k |0\rangle_{n-k-s_t}$, so that the probability that we measure $|1\rangle$ in the output register is $1 - \lambda_j$:

$$\begin{aligned} P(1) &= \left|\Pi_n^{|1\rangle} W_t \cdot |0\rangle_{s_t} |\psi_{t,u}\rangle_k |0\rangle_{n-k-s_t} |0\rangle_1\right|^2 \\ &= \left|\Pi_n^{|1\rangle} |0\rangle_{s_t} |\psi_{t,u}\rangle |0\rangle_{n-k-s_t} \left(\sqrt{\lambda_{t,u}} |0\rangle_1 + \sqrt{1-\lambda_{t,u}} |1\rangle_1\right)\right|^2 \\ &= \left|\sqrt{1-\lambda_{t,u}} |0\rangle |\psi_{t,u}\rangle |0\rangle_{n-k-s_t}\right|^2 = 1 - \lambda_{t,u} \end{aligned}$$

In general, Merlin may send any message $|\mu\rangle_n$. In that case, let us first express $|\mu\rangle_n$ in terms of the eigenvectors of H_t:

$$|\mu\rangle_n = \sum_{x=0}^{2^{s_t}-1} \sum_{u=0}^{2^k-1} \sum_{y=0}^{2^{n-k-s_t}-1} \alpha_{x,u,y} |x\rangle_{s_t} |\psi_{t,u}\rangle_k |y\rangle_{n-k-s_t}$$

What is the probability that we measure the output qubit in state $|1\rangle_1$ if we apply the circuit W_t? This requires only some elementary, albeit tedious algebra:

$$\begin{aligned} P_t(1) &= \left|\Pi_n^{|1\rangle} W_t |\mu\rangle_n |0\rangle_1\right|^2 \\ &= \left|\sum_{x=0}^{2^{s_t}} \sum_{u=0}^{2^k} \sum_{y=0}^{2^{n-k-s_t}} \alpha_{x,u,y} \left(\sqrt{1-\lambda_{t,u}}\right) |x\rangle_{s_t} |\psi_{t,u}\rangle_k |y\rangle_{n-k-s_t} |1\rangle_1\right|^2 \\ &= 1 - \sum_{u=0}^{2^k-1} \lambda_{t,u} \sum_{x=0}^{2^{s_t}} \sum_{y=0}^{2^{n-k-s_t}} |\alpha_{x,u,y}|^2 = 1 - \sum_{u=0}^{2^k-1} \lambda_{t,u} \left|\Pi_{n,s_t}^{|\psi_{t,u}\rangle} |\mu\rangle_n\right|^2 \\ &= 1 - \sum_{u=0}^{2^k-1} \lambda_{t,u} \langle\mu|_n \Pi_{n,s_t}^{|\psi_{t,u}\rangle} |\mu\rangle_n \\ &= 1 - \langle\mu|_n \sum_{u=0}^{2^k-1} \lambda_{t,u} \left(I^{s_t} |\psi_{t,u}\rangle_k \langle\psi_{t,u}|_k I^{n-k-s_t}\right) |\mu\rangle \\ &= 1 - \langle\mu|_n I^{\otimes s_t} \otimes \left(\sum_{u=0}^{2^k-1} \lambda_{t,u} |\psi_{t,u}\rangle_k \langle\psi_{t,u}|_k\right) \otimes I^{\otimes n-k-s_t} |\mu\rangle_n \\ &= 1 - \langle\mu|_n I^{\otimes s_t} \otimes M_t \otimes I^{\otimes n-k-s_t} |\mu\rangle_n = 1 - \langle\mu|_n H_t |\mu\rangle_n \end{aligned}$$

The only step that remains is to describe the full circuit. The subcircuit W_1 will put its output in qubit $n+1$, subcircuit W_2 will put its output in qubit $n+2$, and so forth. If we measure one of these output qubits at random and accept iff we measure 1, then we accept with the desired probability. To this end, the circuit from section 3.0.2 is deployed to select one of the outputs registers for measurement. We skip the complete derivation, because we have already shown that the result is identical to tossing a die with T sides and measuring the corresponding output qubit.

$$\begin{aligned}\Pr(1) =& \sum_{t=1}^{T} \Pr[\text{measure qubit } n+t] \cdot \Pr_t[1] \\ =& \frac{1}{T} \sum_{t=1}^{T} 1 - \langle\mu|_n H_t |\mu\rangle_n \\ =& 1 - \frac{1}{T} \langle\mu|_n \sum_{t=1}^{T} H_t |\mu\rangle_n = 1 - \frac{1}{T} \langle\mu|_n H |\mu\rangle_n\end{aligned}$$

Part 1: Completeness. Suppose that H has a small eigenvalue $\lambda \leq a$, corresponding to a unit eigenvector $|\Psi\rangle_n$. Then Merlin can send us $|\mu\rangle_n = |\Psi\rangle_n$, and we will conclude *"Yes, there is a small eigenvalue"* with probability $1 - \frac{\lambda}{T} \geq 1 - \frac{a}{T}$.

Part 2: Soundness. If all eigenvalues are greater than b, then by Equation 3.14, any message sent by Merlin can make us accept with probability at most $1 - \frac{b}{T}$. Using parallel repetition, we can amplify this gap to any constant we like with a number of gates that is polynomial in $\frac{1}{b-a}$. □

To prove that the Local Hamiltonian problem is QMA-Complete, we will need several more theorems about linear algebra. The exposition of the proof here closely follows that of Aharonov et al. [63].

Definition 5 Angle between subspaces
Let V and W be two subspaces of a vector space. Then the angle $\Theta(V, W)$ between them is the unique real number in $[0, \frac{1}{2}\pi]$ such that

$$\cos\Theta(V, W) = \max_{\substack{|v\rangle \in V \\ |w\rangle \in W}} |\langle v|w\rangle| \tag{3.18}$$

The angle between subspaces has some subtlety. For example, by this defini-

tion, the angle between the xy plane and the xz plane is 0, because they share the x vector. Two subspaces have a non-trivial intersection iff their angle is 0, and are orthogonal iff their angle is $\frac{1}{2}\pi$. We also have $|\langle v|w\rangle| \leq \cos\Theta(V,W)$ for all $|v\rangle \in V, |w\rangle \in W$ and $\cos^2\Theta(V,W) = \max |v\rangle \in V\colon \langle v|\Pi_W|v\rangle$. The notation $A \geq \lambda$, where A is an operator, means $\forall |\phi\rangle : \langle\phi|A|\phi\rangle \geq \lambda$, so $A \geq 0$ means that A is positive semidefinite, for example. If B is an operator, then $A \geq B$ means $\forall |\phi\rangle : \langle\phi|A|\phi\rangle \geq \langle\phi|B|\phi\rangle$, so $A \geq \lambda$ is equivalent to $A \geq \lambda I$. This notation obeys all the obvious inference rules, such as $\lambda A \geq zB$ implies $\lambda A - zB \geq 0$. Lastly, if $A \geq 0$ is an operator with kernel V, and no nonzero eigenvalue of A is smaller than λ, then $A + \lambda\Pi_V \geq \lambda$.

Theorem 40. *Let V and W be two subspaces of a vector space, and Π_V and Π_W the projections onto those spaces. Denote with θ the angle between the spaces. Then $\Pi_V + \Pi_W \leq 1 + \cos\theta$.*

Proof. Let $|c\rangle$ be an eigenvector of $\Pi_V + \Pi_W$ with $(\Pi_V + \Pi_W)|c\rangle = \lambda|c\rangle$. Since Π_V and Π_W are projections, and projections are non-negative operators, we must have $\lambda \geq 0$ by Theorem 35. Then we can (uniquely) express the vectors $\Pi_V|c\rangle$ and $\Pi_W|c\rangle$ as $\Pi_V|c\rangle = x|a\rangle$ and $\Pi_W|c\rangle = y|b\rangle$, with x, y nonnegative real numbers and $|a\rangle$ and $|b\rangle$ unit vectors. Then $x|a\rangle + y|b\rangle = \lambda|c\rangle$. Then we have $\lambda = x^2 + y^2$ and $\lambda^2 = x^2 + y^2 + 2xy\Re\langle a|b\rangle$:

$$\lambda = \langle c|(\Pi_V + \Pi_W)|c\rangle = \langle c|\Pi_V|c\rangle + \langle c|\Pi_W|c\rangle \tag{3.19}$$

$$= \langle c|\Pi_V\Pi_V|c\rangle + \langle c|\Pi_W\Pi_W|c\rangle = |\Pi_V|c\rangle|^2 + |\Pi_W|c\rangle|^2 = x^2 + y^2 \tag{3.20}$$

$$\lambda^2 = (\lambda\langle c|)(\lambda|c\rangle) = (x\langle a| + y\langle a|)(x|a\rangle + y|b\rangle) = x^2 + y^2 + 2xy\Re\langle a|b\rangle \tag{3.21}$$

Next, we derive that $(1+\delta)\lambda - \lambda^2 \geq 0$, where $\delta = |\Re\langle a|b\rangle| \leq \cos\theta$.

$$(1+\delta)\lambda - \lambda^2 = \lambda + \delta\lambda - \lambda^2 = x^2 + y^2 + \delta x^2 + \delta y^2 - x^2 - y^2 - 2xy\delta \tag{3.22}$$

$$= \delta(x^2 + y^2 - 2xy) = \delta(x-y)^2 \geq 0 \tag{3.23}$$

The last inequality is derived from the fact that both δ and $(x+y)^2$ are nonnegative. This almost completes the proof. Just one last step!

$$0 \leq (1+\delta)\lambda - \lambda^2 \text{ so } \lambda^2 \leq (1+\delta)\lambda \text{ so } \lambda \leq 1 + \delta \leq 1 + \cos\theta \tag{3.24}$$

□

Good, next theorem!

Theorem 41. *Let $A, B \geq 0$ be two positive semidefinite operators with kernels V, W, with θ the angle between the two kernels. Suppose that no nonzero eigenvalue of A or B is smaller than λ. Then*

$$A + B \geq 2\lambda \sin^2\left(\frac{1}{2}\theta\right) \tag{3.25}$$

Proof. Clearly we have $A+\lambda\Pi_V \geq \lambda$ and $B+\lambda\Pi_W \geq \lambda$, which gives $A \geq \lambda(I-\Pi_V)$ and $B \geq \lambda(I - \Pi_W)$. So $A + B \geq \lambda(I - \Pi_V) + \lambda(I - \Pi_W)$. We will show that $(I - \Pi_V) + (I - \Pi_W) \geq 2\sin^2(\frac{1}{2}\theta)$, which suffices to prove the theorem. We use two ingredients; first, the following gem from mathematics: $2\sin^2\left(\frac{1}{2}\theta\right) = 1-\cos\theta$. Second, Theorem 40 applied to the kernels of A and B.

$$\begin{aligned}
\Pi_V + \Pi_W &\leq 1 + \cos\theta = 2 - 2\sin^2(1/2\theta) && (3.26)\\
\Pi_V + \Pi_W - 2I &\leq -2\sin^2(1/2\theta) && (3.27)\\
(I - \Pi_V) + (I - \Pi_W) &\geq 2\sin^2(1/2\theta) && (3.28)\\
A + B \geq \lambda(I - \Pi_V) + \lambda(I - \Pi_W) &\geq 2\lambda\sin^2(1/2\theta) && (3.29)
\end{aligned}$$

□

Here is another simple lemma.

Theorem 42. *If $H = \sum_i \lambda_i |\psi_i\rangle \langle\psi_i|$ is a Hermitian matrix, m its smallest eigenvalue and $|\phi\rangle$ some unit vector, then $\langle\phi| H |\phi\rangle \geq m$.*

Proof. We can solve this theorem with simple algebraic manipulations. Decompose $|\phi\rangle = \sum_i \alpha_i |\psi_i\rangle$ into the orthonormal base of eigenvectors of H.

$$\langle\phi| H |\phi\rangle = \sum_i \lambda_i |\alpha_i|^2 \geq \sum_i m |\alpha_i|^2 = m \sum_i |\alpha_i|^2 = m \tag{3.30}$$

□

We are now ready to proceed to the main result.

Theorem 43. *The Local Hamiltonian Problem is* **QMA**-*Complete.*

Proof. We will show a construction to take a circuit and produce a local Hamiltonian. Let $U = U_1 \ldots, U_T$ be a series of gates which act on qubit sets $S_1, \ldots, S_T$ respectively. The circuit U acts on $n = w + m$ qubits: there are w qubits in the work space, initialised as $|0\rangle_w$, and there are m qubits in the message from Merlin, $|\mu\rangle_m$. It is promised that this circuit decides with high probability, that is:

$$\text{Either } \exists\, |\mu\rangle_m : \left| \Pi_{n,1}^{|1\rangle} U |\mu\rangle_m |0\rangle_w \right|^2 \geq 1 - \frac{1}{2^n}, \tag{3.31}$$

$$\text{or } \forall\, |\mu\rangle_m : \left| \Pi_{n,1}^{|1\rangle} U |\mu\rangle_m |0\rangle_w \right|^2 \leq \frac{1}{2^n} \tag{3.32}$$

We will create a Hamiltonian $\mathcal{H}$ which has a small eigenvalue if and only if Equation 3.31 is true, and has only large eigenvalues if Equation 3.32 is true. $\mathcal{H}$ consists of three parts:

$$\mathcal{H} = H_{in} + H_{prop} + H_{out} \tag{3.33}$$

The three terms are themselves the sum of many Hermitian operators, each operating on only a small number of qubits. This Hamiltonian will act on $m+w+T$ qubits. The last T qubits will be used to bookkeep time; this register is a counter. At the end of the proof, we will revisit the counter and make sure that it satisfies the locality and normalization constraints.

Let us return to the proof that SAT is **NP**-Complete. The reduction from an arbitrary language L to SAT constructs a Boolean formula which encodes a computation of the Turing Machine. A satisfying assignment to the variables provides a transcript of a succesful, accepting computation. The clause may be interpreted as, *There exists a computation path such that, if the Turing Machine is initialised to the empty tape, and the execution is carried out for a number of steps, then there is a transcript of a computation in which each transition follows the previous one, ending in an accepting configuration.* We will attempt to emulate this proof for the present problem.

The quantum analogue of an assignment which satisfies a formula is a unit vector $|\zeta\rangle_{w+m+T}$ which satisfies $|\mathcal{H}|\zeta\rangle| \leq a$. Hence Merlin will try to find a state $|\zeta\rangle_{w+m+T}$ which minimizes the penalty function $p(|\zeta\rangle) = |\langle\zeta|\mathcal{H}|\zeta\rangle|$ (the minimum of this function is found at an eigenvalue). We will interpret the state $|\zeta\rangle$ as a transcript of a computation by the given circuit and choose $\mathcal{H}$ such that it penalizes Merlin (i.e. it increases p) when he (a) gives a transcript in which the work qubits are not initialised to $|0\rangle_n$, or (b) his transcript does not encode a valid

computation, i.e. the states do not follow from one moment to the next in the way specified by the circuit U, or (c) the computation does not accept. Scenario (a) is covered by H_{in}, scenario (b) is covered by H_{prop} (for *propagation*) and scenario (c) is covered by H_{out}:

$$H_{in} = \sum_{i=1}^{w} \Pi_{n,i}^{|1\rangle_1} \otimes |0\rangle_T \langle 0|_T \tag{3.34}$$

$$H_{prop} = \sum_{t=1}^{T} I \otimes |t\rangle \langle t| + I \otimes |t-1\rangle \langle t-1| - U_t \otimes |t\rangle \langle t-1| - U_t^\dagger |t-1\rangle \langle t| \tag{3.35}$$

$$H_{out} = \Pi_{n,1}^{|0\rangle_1} \otimes |T\rangle_T \langle T| \tag{3.36}$$

$$\tag{3.37}$$

We proceed in three steps. First, we show that the reduction satisfies completeness: If there is a $|\mu\rangle$ which the circuit accepts, then the Hamiltonian has a small eigenvalue. Second, we show that the reduction has soundness: If there is no message $|\mu\rangle$ which the circuit accepts, then the Hamiltonian has no small eigenvalues (we will find a lower bound for its eigenvalues). Third and last, we will reformat the counting register to make sure that (i) all terms are 5-local and (ii) for each term, the eigenvalues are $0 \leq \lambda \leq 1$.

3.1.1 Completeness: A small eigenvalue when the answer is "*yes*"

We will look at the Hamiltonian from Merlin's perspective and ask, for each term H in $\mathcal{H}$: what message $|\mu\rangle_{n+T}$ should he send to minimize $\langle\mu| H |\mu\rangle$? From the outset, Merlin may send an arbitrary state $|\mu\rangle_{n+T}$, where for convenience we distill $|\sigma_t\rangle$ for $0 \leq t \leq T$. Note that $|\sigma_t\rangle$ is not necessarily a unit vector, and in fact may be the zero vector.

$$|\mu\rangle_{n+T} = \sum_{t=0}^{T} \sum_{x=0}^{2^n-1} \alpha_{t,x} |x\rangle_n |t\rangle_T \qquad |\sigma_t\rangle_n = \sum_{x=0}^{2^n-1} \alpha_{t,x} |x\rangle_n \tag{3.38}$$

The idea will be that Merlin's message is a transcript of a succesful computation. Impatient readers may find it in Equation 3.44, but to stress the similarity

between this proof and the original Cook-Levin theorem, it is instructional to derive this fact from the Hamiltonian.

Consider the task of minimizing the first term, $\left|H_{in}\left|\mu\right\rangle_{n+T}\right|$:

$$\left|H_{in}\left|\mu\right\rangle_{n+T}\right| = \left|\sum_{i=1}^{w} \Pi_{n,i}^{\left|1\right\rangle_1} \otimes \left|0\right\rangle_T \left\langle 0\right|_T \left|\mu\right\rangle_{n+T}\right| = \left|\sum_{i=1}^{w} \Pi_{n,i}^{\left|1\right\rangle_1} \vec{\sigma_0}\right| \tag{3.39}$$

Merlin can do this one of two ways. Either he sets $\left|\sigma_0\right\rangle = \vec{0}$, the zero vector, or, if $\left|\zeta\right\rangle_m$ is a certificate that the circuit accepts, he can set $\left|\sigma_0\right\rangle = \frac{1}{\sqrt{T+1}} \left|0\right\rangle_w \left|\zeta\right\rangle_m$. Note that the latter option is an honest transcript: at time $t = 0$, the system is in state $\left|0\right\rangle_n \left|\zeta\right\rangle_m$. We show that the latter is a good strategy, and in fact, that choosing for each $0 \leq t \leq T$: $\left|\left|\sigma_t\right\rangle\right| = \frac{1}{\sqrt{T+1}}$ is a good strategy.

To minimize $\left\langle\mu\right| H_{out} \left|\mu\right\rangle$, Merlin should set $\vec{\sigma_T}$ to

$$\vec{\sigma_T} = \frac{1}{\sqrt{T+1}} U \left|0\right\rangle_w \left|\zeta\right\rangle_m \tag{3.40}$$

$$\text{So } \left|H_{out} \cdot \left|\mu\right\rangle\right| = \left|\frac{1}{\sqrt{T+1}} \Pi_{n,1}^{\left|1\right\rangle} U \left|0\right\rangle_w \left|\zeta\right\rangle_m\right| \leq \frac{\varepsilon}{\sqrt{T+1}} \tag{3.41}$$

Note that this is again an honest transcript of the computation, because at time $t = T$, the system is in the state $U\left|0\right\rangle\left|\zeta\right\rangle$. Of course, Merlin is free to set $\left|\sigma_T\right\rangle = \left|1\right\rangle_1 \left|0\right\rangle_{n-1}$ instead of $\left|\sigma_T\right\rangle = U\left|0\right\rangle\left|\zeta\right\rangle$, minimizing the penalty function, but that would be cheating, and since the computation is accepting, there is no need for that.

Consider, lastly, H_{prop}:

$$\left\langle\mu\right| H_{prop} \left|\mu\right\rangle = \frac{1}{2} \sum_{t=1}^{T} \left\|\vec{\sigma_t}\right\|^2 + \left\|\vec{\sigma}_{t-1}\right\|^2 - \left\langle\sigma_t\right| U_t \left|\sigma_{t-1}\right\rangle - \left\langle\sigma_{t-1}\right| U_t^\dagger \left|\sigma_t\right\rangle \tag{3.42}$$

$$= \sum_{t=1}^{T} \frac{1}{2}\left\|\vec{\sigma_t}\right\|^2 + \frac{1}{2}\left\|\vec{\sigma}_{t-1}\right\|^2 - \Re\left\langle\sigma_{t-1}\right| U_t^\dagger \left|\sigma_t\right\rangle \tag{3.43}$$

To maximize this term, Merlin should maximize $\left\langle\sigma_{t-1}\right| U^\dagger \left|\sigma_t\right\rangle$ for $1 \leq t \leq T$. This is easily done by setting $\left|\sigma_t\right\rangle = U_t \left|\sigma_{t-1}\right\rangle$, so that $\left\langle\sigma_{t-1}\right| U_t^\dagger \left|\sigma_t\right\rangle =$

$\langle\sigma_{t-1}| U^\dagger U |\sigma_{t-1}\rangle = 1$. This is an honest transcript, as the state at time t is related to the state at time $t-1$ by $|\sigma_t\rangle = U|\sigma_{t-1}\rangle$. Hence, all told, Merlin may send the following state:

$$|\mu\rangle_{n+T} = \frac{1}{\sqrt{T+1}} \sum_{t=0}^{T} U_t \dots U_1 |\zeta\rangle_m |0\rangle_n |t\rangle \tag{3.44}$$

Let us verify that this strategy works. We know that $\langle\mu| H_{in} |\mu\rangle = 0$ and $\langle\mu| H_{out} |\mu\rangle = \frac{\varepsilon}{T+1}$. The terms of H_{prop} contribute nothing:

$$\langle\mu| H_{prop} |\mu\rangle = \frac{1}{T+1} \cdot \frac{1}{2} \sum_{t=1}^{T} 2 - 2\Re \langle\sigma_{t-1}| U^\dagger |\sigma_t\rangle = 0 \tag{3.45}$$

Great! We can express $\langle\mu| \mathcal{H} |\mu\rangle$ at long last:

$$\begin{aligned}
\langle\mu| \mathcal{H} |\mu\rangle &= \langle\mu| (&& H_{in} && + H_{out} && + H_{prop}) |\mu\rangle && (3.46)\\
&= && \langle\mu| H_{in} |\mu\rangle && + \langle\mu| H_{out} |\mu\rangle && + \langle\mu| H_{prop} |\mu\rangle && (3.47)\\
&= && 0 && + \frac{\varepsilon}{T+1} && + 0 && (3.48)\\
&= && \frac{\varepsilon}{T+1} && && && (3.49)
\end{aligned}$$

By Theorem X we know that there must exist an eigenvalue $m \leq \langle\mu| \mathcal{H} |\mu\rangle = \frac{\varepsilon}{T+1}$. Hence Merlin can send $|\mu\rangle$, or he can send the smallest eigenvector; both will convince Arthur.

3.1.2 Soundness: Lower bound when the answer is *"no"*

Suppose that the circuit rejects all certificates with high probability. Denote with ε the probability with which the circuit accepts the "best" certificate:

$$\varepsilon = \max_{|\mu\rangle_m \in \mathcal{B}^{\otimes m}} \left| \Pi_1^{|1\rangle} U |\mu\rangle_m |0\rangle_w \right|^2 \leq \frac{1}{2^T} \tag{3.50}$$

We will have to show that in this case, the Hamiltonian $\mathcal{H}$ produced by the construction above has only large eigenvalues. To this end, we divide the Hamiltonian in two parts, $\mathcal{H} = H_1 + H_{prop}$, with $H_1 = H_{in} + H_{out}$. Having written $\mathcal{H}$ as the sum of two operators, we may invoke Theorem 41 to obtain a lower bound on the eigenvalue of $\mathcal{H}$.

The plan is as follows. Theorem 41 will give us a lower bound for the eigenvalues of $\mathcal{H}$ when we (i) find a λ which lower bounds the nonzero eigenvalues of H_1 and H_{prop} and (ii) find the angle between the subspaces of the kernels of H_1 and H_{prop}. Let $V = \ker(H_1)$ and $W = \ker(H_{prop})$. Then if we can find the angle $\Theta(V, W)$, we have a good lower bound:

$$\mathcal{H} = H_1 + H_{prop} \geq 2\lambda \sin^2\left(\frac{\theta(V,W)}{2}\right) \tag{3.51}$$

First, the eigenvalues of H_1 and H_{prop}. The operator H_1 is a projection, so its only eigenvalue is 1. For H_{prop}, it turns out the operator is better represented in a rotated basis. The rotation R is

$$R = \sum_{t=0}^{T} U_t \cdots U_1 \otimes |t\rangle \langle t| \tag{3.52}$$

This operation is unitary because it is a block matrix in which each block is unitary (the t-th block is $U_t \cdots U_1$). Therefore, we can change the basis through which we view H_{prop}, and obtain a new Hamiltonian $\tilde{H}_{prop}$. It suffices to analyze this new Hermitian, because eigenvalues are invariant under a change of basis.

$$\tilde{H}_{prop} = R^\dagger H_{prop} R = \sum_{t=1}^{T} I \otimes (|t\rangle \langle t| + |t-1\rangle \langle t-1| - |t\rangle \langle t-1| - |t-1\rangle \langle t|) \tag{3.53}$$

Now $\tilde{H}_{prop} = I \otimes A$ where $A = I - B$ and B is as follows (for $T = 4$):

$$A = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} \end{bmatrix} = I - \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} = I - B \tag{3.54}$$

The smallest eigenvalue of A is at least $\frac{1}{2(T+1)^2}$. This is a result from the theory of random walks [63]. Therefore we take $\lambda = \frac{1}{2(T+1)^2}$, as it lower bounds the eigenvalues of both $\tilde{H}_{prop}$ and $\tilde{H}_1$.

For (ii) we will express all operators in the rotated basis, with $\tilde{H}_{in} = R^\dagger H_{in} R = H_{in}$ and $\tilde{H}_{out} = R^\dagger H_{out} R = U^\dagger H_{out} U \otimes |T\rangle\langle T|$. Let $V = \ker(\tilde{H}_1) = \ker(\tilde{H}_{in} + \tilde{H}_{out})$ and $W = \ker(\tilde{H}_{prop})$. Then

$$\ker(\tilde{H}_1) = |0\rangle_w \otimes \mathcal{B}^{\otimes w} \otimes |0\rangle \oplus \mathcal{B}^{\otimes w+m} \otimes (|1\rangle \oplus \cdots \oplus |T-1\rangle) \tag{3.55}$$

$$\oplus U^\dagger \cdot |1\rangle_1 \otimes \mathcal{B}^{\otimes n+m-1} \otimes |T\rangle \tag{3.56}$$

$$\ker(\tilde{H}_{prop}) = \mathcal{B}^{\otimes n} \otimes \sum_{t=0}^{T} |t\rangle \tag{3.57}$$

Here the last register in each term pertains to the clock. The projection onto $V = \ker(\tilde{H}_1)$ breaks into three parts, denoted Π_1, Π_2, Π_3:

$$\Pi_V = \underbrace{I^m |0\rangle_w \langle 0|_w |0\rangle\langle 0|}_{\Pi_1} + \underbrace{I^{\otimes n} \sum_{t=1}^{T-1} |t\rangle\langle t|}_{\Pi_2} + \underbrace{U^\dagger \Pi^{|1\rangle} U |T\rangle\langle T|}_{\Pi_3} \tag{3.58}$$

Instead of finding the angle $\theta(V,W)$, we will find $\cos^2\theta(V,W)$. Any vector $|\psi\rangle \in W$ can be represented as $|\psi\rangle = \frac{1}{\sqrt{T+1}} |\phi\rangle \sum_{t=0}^{T} |t\rangle$ for some $|\phi\rangle \in \mathcal{B}^{\otimes n}$. Note that $\langle\psi| \Pi_2 |\psi\rangle = \frac{T-1}{T+1}$.

$$\cos^2\theta(V,W) = \max_{\substack{|\psi\rangle \in W \\ |\alpha\rangle \in V}} |\langle\psi|\alpha\rangle|^2 = \max_{|\psi\rangle \in W} \langle\psi| \Pi_V |\psi\rangle \tag{3.59}$$

$$= \max_{|\phi\rangle \in \mathcal{B}^{\otimes n}} \frac{1}{T+1} \langle\phi| \left(\Pi^{|0\rangle_w} + U^\dagger \Pi_1^{|1\rangle} U \right) |\phi\rangle + \frac{T-1}{T+1} \tag{3.60}$$

This forces us into a brief aside in which we estimate the largest eigenvalue of the projection $\Pi^{|0\rangle_w} + U^\dagger \Pi^{|1\rangle} U$. Since this is a sum of two projections, we can invoke Theorem 40 if we can find the angle $\Theta(X,Y)$ between the spaces onto which they project. We have $X = \mathcal{B}^{\otimes m} |0\rangle_w$ and $Y = U^\dagger \cdot |1\rangle \otimes \mathcal{B}^{\otimes n-1}$. Again, we will not estimate $\Theta(X,Y)$ directly, but we will find $\cos^2\Theta(X,Y)$, which turns out to be exactly ε, the maximum probability with which our circuit accepts any certificate. A vector from X can be expressed as $|\mu\rangle_m |0\rangle_w$ for some $|\mu\rangle_m$.

$$\cos^2(X,Y) = \max_{\substack{|x\rangle\in X\\ |y\rangle\in Y}} |\langle x|y\rangle|^2 \tag{3.61}$$

$$= \max_{|\mu\rangle\in\mathcal{B}^{\otimes m}} \langle 0|_w \langle\mu|_m U^\dagger \Pi^{|1\rangle} U |\mu\rangle_m |0\rangle_w \tag{3.62}$$

$$= \max_{|\mu\rangle\in\mathcal{B}^{\otimes m}} \left|\Pi^{|1\rangle} U |\mu\rangle_m |0\rangle_w\right|^2 = \varepsilon \tag{3.63}$$

We have $\cos(X,Y) = \sqrt{\varepsilon}$. Theorem 40 translates this cosine into an upper bound: $\Pi^{|0\rangle_w} + U^\dagger \Pi^{|1\rangle} U \geq 1 + \cos\theta = 1 + \sqrt{\varepsilon}$. We resume finding $\cos^2(V,W)$:

$$\cos^2(V,W) = \frac{T-1}{T+1} + \frac{1+\sqrt{\varepsilon}}{T+1} = 1 - \frac{1-\sqrt{\varepsilon}}{T+1} \tag{3.64}$$

We're almost done. For the lower bound of $\mathcal{H}$, we have to reason about $\sin^2(1/2\Theta(V,W))$.

$$\sin^2(\Theta(V,W)) = 1 - \cos^2(\Theta(V,W)) = \frac{1-\sqrt{\varepsilon}}{T+1} \tag{3.65}$$

$$\sin^2(\frac{1}{2}\Theta(V,W)) \geq \frac{1}{4}\sin^2(\Theta(V,W)) \tag{3.66}$$

We are now ready to substitute $\sin^2(\frac{1}{2}\Theta)$ and λ into Equation 3.51:

$$\mathcal{H} \geq \frac{1}{2}\lambda \sin^2(\Theta(V,W)) \geq \frac{1}{2}\cdot\frac{1}{2(T+1)^2}\cdot\frac{1-\sqrt{\varepsilon}}{T+1} = \frac{1-\sqrt{\varepsilon}}{4(T+1)^3} \tag{3.67}$$

3.1.3 Realization of the counter

We now turn our attention to ensuring that all terms are 5-local by specifying how the operators that act on the clock space are implemented. For example, $H_{out} = |0\rangle_1 \langle 0|_1 \otimes I^{n-1} \otimes |T\rangle\langle T|$ acts on one qubit and the clock register. We could, of course, write T in binary and use the operator $H_{out} = |0\rangle_1 \langle 0|_1 \otimes I^{\otimes n-1} \otimes |T\rangle_{\log(T)} \langle T|_{\log(T)}$, but then the operator acts on $\log(T)+1$ qubits, and not on 5 qubits.

Instead, we will implement the clock register as numbers in unary representation using $T+1$ qubits. For example, $t=0$ will be represented as $|0\rangle_{T+1} = |100\cdots00\rangle_{T+1}$ and $t=3$ is represented as $|3\rangle_{T+1} = |111100\cdots00\rangle_{T+1}$. The clock operators are represented as follows:

$$|t\rangle\langle t| \mapsto I^{\otimes t-1} \otimes |110\rangle_3\langle 110|_3 \otimes I^{\otimes T-t-1} \tag{3.68}$$
$$|t\rangle\langle t-1| \mapsto I^{\otimes t-1} \otimes |110\rangle_3\langle 100|_3 \otimes I^{\otimes T-t-1} \tag{3.69}$$
$$|t-1\rangle\langle t| \mapsto I^{\otimes t-1} \otimes |100\rangle_3\langle 110|_3 \otimes I^{\otimes T-t-1} \tag{3.70}$$
$$|t-1\rangle\langle t-1| \mapsto I^{\otimes t-1} \otimes |100\rangle_3\langle 100|_3 \otimes I^{\otimes T-t-1} \tag{3.71}$$

For the cases $t=0$ and $t=T$ we make an exception, dropping the first and last qubit, respectively, so as not to refer to qubits which do not exist. So $|0\rangle\langle 0| \mapsto |10\rangle\langle 10|$ and $|T\rangle\langle T| \mapsto I^{\otimes T-1} \otimes |11\rangle\langle 11|$. So for example, H_{out} will be implemented as $|0\rangle_1\langle 0|_1 \otimes I^{\otimes n-1} \otimes I^{\otimes T-1} |11\rangle_2\langle 11|_2$, which acts non-trivially on 3 qubits. The t-th term of $\tilde{H}_{prop}$ is implemented as

$$\tilde{H}_{prop} = I^{\otimes n} \otimes I^{\otimes t-1} \otimes (|110\rangle\langle 110| + |100\rangle\langle 100| - |100\rangle\langle 110| - |110\rangle\langle 100|) \otimes I^{\otimes T-t-1} \tag{3.72}$$

The argument above for the completeness of the protocol still goes through with this scheme. Merlin will simply have to send the state in Equation 3.44 with $|t\rangle$ represented in unary. However, the argument for soundness does not go through anymore, because Merlin may send states whose clock register is not unary (for example, the state $|0\rangle_n |00100\rangle_5$ is not affected by any Hermitian). To rectify this, we add one last term, H_{clock}, to the Hamiltonian which penalizes states which do not represent their clock register as a unary number:

$$H_t = \sum_{t=0}^{T} I^{\otimes n} \otimes I^{\otimes t} |01\rangle_2\langle 01|_2 \otimes I^{\otimes T-1} \tag{3.73}$$

Then the final Hamiltonian is as follows:

$$\mathcal{H} = H_{in} + H_{prop} + H_{out} + H_{clock} \tag{3.74}$$

In this setting, the soundness argument goes through because we take $H_1 = H_{in} + H_{out} + H_{clock}$, whose eigenvalues are all 1, and whose kernel does not intersect the kernel of H_{prop}, and which has the same angle to it. □

3.1.4 Some remarks about the proof

This proof is sometimes called the quantum Cook-Levin theorem because it demonstrates the "quantum **NP**-Completeness" of a quantum analogue of the SAT problem. The connection is very deep. First, the $MAX - k - SAT$ problem can be reduced to the k-local Hamiltonian problem in a way that shows that the k-local Hamiltonian problem is a natural generalization of SAT. Second, the structure of the two proofs (the proof of the Cook-Levin theorem and the proof above) are very similar: they show how to interpret a computation as the specification for how a system evolves over time, and show that the question of whether a certain input-output combination exists relative to this time evolution can be translated into a satisfiability problem.

Lastly, they map a satisfying assignment in one problem to a satisfying assignment in another. This property is much stronger than is necessary for a reduction. Any reduction preserves membership of instances, but the Cook-Levin reduction also preserves the structure of the solution space. That is, whereas a reduction need only be a function f for which $x \in L \iff f(x) \in \text{SAT}$, the Cook-Levin Theorem gives us a way to transform, in polynomial time, a certificate for a non-deterministic machine into a satisfiable assignment to ϕ. Moreover, the map between satisfying assignments and accepted certificates is bijective, so the number of solutions to the problems is preserved. A reduction which preserves the number of solutions is called a *parsimonious* reduction.[3]

Our reduction to the Local Hamiltonian Problem gives the following mapping between satisfying certificates for a circuit and the right eigenvectors for the Hamiltonian:

$$|\mu\rangle_m \mapsto \sum_{t=0}^{T} U_t \cdots U_1 |\mu\rangle_m |0\rangle_w |t\rangle \tag{3.75}$$

This transformation is easily implemented in a quantum circuit. One simply puts the time register in the superposition $\sum_{t=0}^{T} |t\rangle$ (for example, using a Fourier transform) and then acts on the system with the operator $|\mu\rangle_m |0\rangle_w |t\rangle \mapsto U_t \cdots U_1 |\mu\rangle |0\rangle |t\rangle$, in other words, we execute the first t gates of the circuit.

[3]The concept of parsimonious reducibility is not always applicable, as not all complexity classes are defined with non-deterministic Turing Machines in mind, and of course because not all languages are accepted by polynomial-time nondeterministic Turing Machines. The concept, to the best of our knowledge, pertains only to reductions between (the decision versions of) search problems, i.e. to **NP** problems and, by the reduction we have just given, to **QMA** problems.

The reduction in Kitaev's proof, then, preserves the structure of the underlying solution space via a linear map, so it is a parsimonious reduction in the sense just described. It is a longstanding open question whether parsimonious reductions exist between all **NP**-Complete problems. In 2008, Goldreich mentioned that parsimonious reductions exist between all known **NP**-Complete languages [5]. To the best of the author's knowledge, that statement still holds at the time of writing.

This open problem is not to be confused with a conjecture, by Berman and Hartmanis, called the isomorphism conjecture [6], which posits that all **NP**-Complete languages are interreducible by polynomial-time invertible bijections, which is to say: for every pair L, K of **NP**-Complete languages, there is a polynomial-time Karp reduction f from L to K such that f is bijective and f^{-1} can be computed in polynomial time. This conjecture only pertains to instances of the languages and not to the solution spaces of machines that recognize them. Both of these conjectures are likely to remain unsolved for some time, as they require (if they are true) proving $\mathbf{P} \neq \mathbf{NP}$ as a first step, as the isomorphism conjecture implies $\mathbf{P} \neq \mathbf{NP}$.

3.2 A relativized version of the Local Hamiltonian problem

The SATISFIABILITY problem captures **NP**-Completeness in the unrelativized world. In some sense the fact that SATISFIABILITY is **NP**-Complete is a non-relativizing statement: there are oracles $\mathcal{O}$, relative to which $\mathbf{NP}^{\mathcal{O}}$ is a class of languages with complete problems, but SATISFIABILITY is not one of those complete problems. For certain purposes, this is an immensely useful fact: apparently SATISFIABILITY captures some unique fact about computation in the real world that is lost when the Turing Machine gets access to an oracle.

In other contexts, however, it is useful to have a natural **NP**-Complete problem even in a relativized world. Fortunately, if SATISFIABILITY is modified to allow oracle predicates in the formula, then it is **NP**-Complete again. Whereas usually only clauses of the form $(x_1 \vee x_5 \vee \neg x_9)$ are allowed, we now allow clauses of the form $(x_1 \vee x_5 \vee \neg\mathcal{O}(x_2, x_1, x_4, x_8, x_1))$, where $\mathcal{O}(x_2, x_1, x_4, x_8, x_1)$ evaluates to true iff the binary string $q \in \{0,1\}^5$ of five characters obtained by concatenating the literals x_2, x_1, x_4, x_8, x_1, is in $\mathcal{O}$, i.e. if $q \in \mathcal{O}$.

Theorem 44. *For every oracle $\mathcal{O} \subseteq \{0,1\}^\star$, the problem* SATISFIABILITY$^{\mathcal{O}}$ *is* $\mathbf{NP}^{\mathcal{O}}$*-Complete, where* SATISFIABILITY$^{\mathcal{O}}$ *is the* SATISFIABILITY *problem, augmented to allow oracle terms to appear in the clauses.*

In this spirit, we now show how to modify the LOCAL HAMILTONIAN problem to account for the presence of oracles. This problem will be complete for **PromiseQMA**, instead of the class **QMA** studied in Chapter 2. The class **QMA** is neither known nor expected to have Complete problems.

The unrelativized version of **QMA** has a natural complete promise problem that we know and love: the LOCAL HAMILTONIAN PROBLEM. Kitaev proves that this problem is in **PromiseQMA**, and then proves that it is hard for **PromiseQMA**, reducing from the trivially Complete DOES-THIS-QMA-MACHINE-ACCEPT-A-STATE-problem.

We give an extremely brief sketch of the original proofs, first of **QMA**-Hardness and then **QMA**-Completeness, containing only the parts that we will need to change to account for oracles.

Theorem 45. *As a promise problem, the* LOCAL HAMILTONIAN PROBLEM *is* PROMISEQMA*-Hard.*

Proof. Given is a circuit and the promise that either the circuit either accepts some state $|\psi\rangle$ confidently, or it rejects them all confidently, but the maximum acceptance probability is never $1/2$ or something silly like that. The circuit has a ancilla qubits that are initialised to $|0\rangle_a$, an answer qubit that should be $|1\rangle$ at the end, and a third register for the input state, $|\psi\rangle$.

The Hamiltonian that is constructed will encode the computation history, a transcript, of a computation by the circuit. It is the sum of three parts. The first two parts "penalize" the Hamiltonian if the transcript encodes a computation that does not end up accepting the input, or if the transcript "cheats" by initializing the ancilla bits to some other value than $|0\rangle_a$:

$$H_{in} = \sum_{i=0}^{a-1} I^i \otimes |1\rangle\langle 1| \otimes I \qquad\qquad H_{out} = I \otimes |1\rangle\langle 1| \tag{3.76}$$

The important part, which we will have to revisit, is the part where for each gate in the circuit, the Hamiltonian has a term to ensure that the transcript shows a computation which executes this gate faithfully. If the t-th gate is U, then add the following term, H_t, to the Hamiltonian:

$$H_t = \frac{1}{2} I \otimes (|t\rangle\langle t| + |t-1\rangle\langle t-1|) - \frac{1}{2} U |t\rangle\langle t-1| - \frac{1}{2} U^\dagger |t\rangle\langle t-1| \tag{3.77}$$

Then a bunch of mathematics happens, and *tada!* You can find good bounds α and β, which are separated by a mile, $\alpha < \beta$. If the computation was rejecting, then the Hamiltonian has all eigenvalues $> \beta$, whereas if it was rejecting, then it has an eigenvalue $< \alpha$, which proves the theorem. □

Next we rehearse how a quantum circuit evaluates a Local Hamiltonian.

Theorem 46. *The* LOCAL HAMILTONIAN *problem is in* **PromiseQMA**.

Proof. For each non-oracle term H, find all eigenvalues of H, say they are $H = \sum \lambda_i |\psi_i\rangle \langle\psi_i|$. Then build a circuit W which acts as follows on this basis of eigenvalues with an ancilla qubit:

$$W |\psi_i\rangle |0\rangle = |\psi_i\rangle \otimes \left(\sqrt{\lambda_i} |0\rangle + \sqrt{1-\lambda_i} |1\rangle\right) \tag{3.78}$$

(We will need the proof up to here) For each circuit, use a different ancilla qubit. Lastly, measure one of the ancilla qubits at random. Then a bunch of mathematics happens, and it all works out neatly: the circuit accepts with an admissible error rate. □

Alright, we are ready to get into this oracle business! For an oracle $\mathcal{A}$, the class **Promise-QMA**$^{\mathcal{A}}$ is the set of tuples $\langle L_{yes}, L_{no}\rangle$ of languages with a yes-part L_{yes} and a no-part L_{no}, disjoint, $L_{yes} \cap L_{no} = \emptyset$ for which there is a uniform quantum circuit family $\{C_x^{\mathcal{A}}\}$ with the following behaviour,

- If $x \in L_{yes}$ then there is a quantum state $|\psi\rangle$ such that $P[C_x^{\mathcal{A}} |0\rangle |\psi\rangle |0\rangle = 1] \geq 2/3$, and
- If $x \in L_{no}$ then for all quantum states $|\psi\rangle$, $P[C_x^{\mathcal{A}} |0\rangle |\psi\rangle |0\rangle = 0] \geq 2/3$.

By $C_x^{\mathcal{A}}$ we mean a circuit whose gate set includes $\{\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots\}$. This gives our quantum algorithms oracle access to $\mathcal{A}$. Here A_n is a gate on $n+1$ qubits which operates as follows on the basis vectors:

$$\mathcal{A} |x\rangle_n |a\rangle_1 = |x\rangle_n |a \oplus \mathcal{A}(x)\rangle_1 = \begin{cases} |x\rangle_n |a\rangle & \text{If } x \notin \mathcal{A} \\ |x\rangle_n |a \oplus 1\rangle & \text{If } x \in \mathcal{A} \end{cases} \tag{3.79}$$

Theorem 47. *For any oracle $\mathcal{A}$, the promise problem* Local Hamiltonian$^{\mathcal{A}}$ *is Hard for* **Promise-QMA**$^{\mathcal{A}}$.

Proof. The input in the Hardness reduction is a series of gates, including some oracle gates. Kitaev has already done the heavy lifting and showed what we are aiming for: Given an oracle gate $\mathcal{A}_k$, we would like to add the term

$$H_t = \frac{1}{2} I \otimes (|t\rangle\langle t| + |t-1\rangle\langle t-1|) - \frac{1}{2}\mathcal{A}_k \otimes |t\rangle\langle t-1| - \frac{1}{2}\mathcal{A}_k^\dagger \otimes |t-1\rangle\langle t| \quad (3.80)$$

to the Hamiltonian. This seems very difficult; one might think that we would have to write out a 2^k by 2^k matrix, where k is non-constant and possibly very large. Instead, we will cheat: we will write something to the tune of "*and here is where we would like to apply the oracle gate to qubits* $3, 5, 4, 13$ *and* 23", it being understood that the term we have in mind is the one in Equation 3.80. This is similar to when we relativize the SAT problem and get a formula that looks like

$$\exists y \in \{0,1\}^n \colon (y_1 \vee y_2 \vee \neg y_3) \wedge (y_2 \vee \neg\mathcal{A}(x_1, x_2, y_1, y_3, y_5) \vee y_5) \quad (3.81)$$

$$\wedge (x_1 \vee y_6 \vee \mathcal{A}(x_3, x_2, y_1, y_2, y_3)) \wedge \ldots \quad (3.82)$$

□

So that's settled, then: it's hard. How do you evaluate this Hamiltonian, when you get it as an input? This is the part where we prove Local Hamiltonian$^{\mathcal{A}} \in$ **Promise-QMA**$^{\mathcal{A}}$. With some effort finding an oracle's eigenvalues, the proof of Kitaev goes through as usual.

Theorem 48. *For every oracle $\mathcal{A} \subseteq \{0,1\}^\star$, the problem* Local Hamiltonian$^{\mathcal{A}}$ *has a* **PromiseQMA**$^{\mathcal{A}}$ *protocol.*

Proof. In Kitaev's proof, the algorithm must first find *all* the eigenvalues of the Hermitians in the input. Clearly, since $\mathcal{A}$ may be queried on an arbitrary number of qubits, say k qubits, and k may be very large and non-constant, we do not have time to write out the 2^k by 2^k matrix and compute the eigenvectors that way. Fortunately, that is easily circumvented by querying the oracle. Let $\mathcal{A}(x) = 1$ if $x \in \mathcal{A}$ and $\mathcal{A}(x) = 0$ otherwise, as described above. Then the oracle acts on the computational basis as:

$$\mathcal{A}|x\rangle|a\rangle = |x\rangle|a \oplus \mathcal{A}(x)\rangle \quad (3.83)$$

The eigenvectors of the gate $\mathcal{A}$ are $\{|x\rangle|a\rangle \,|\, x \notin \mathcal{A}\} \cup \{|x\rangle|+\rangle \,|\, x \in \mathcal{A}\}$ with eigenvalue 1 and $\{|x\rangle|-\rangle \,|\, |x\rangle \in \mathcal{A}\}$ with eigenvalue -1. This is almost what we

want; we wanted to know the eigenvectors to H_t, from Equation 3.80 so that we can build a circuit which behaves according to Equation 3.78. Fortunately, the eigenvalues of H_t can be described as follows (here H_t is the Hermitian term from Equation 3.80 and H is the Hadamard gate):

$$\text{Let } |\tau_0\rangle = \frac{1}{\sqrt{2}}\left(|t\rangle + |t-1\rangle\right) \qquad |\tau_1\rangle = \frac{1}{\sqrt{2}}\left(|t\rangle - |t-1\rangle\right) \tag{3.84}$$

$$\text{Then } H_t \left|x\right\rangle H \left|i\right\rangle_1 \left|\tau_k\right\rangle \equiv \lambda(x,i,k) \left|x\right\rangle H \left|i\right\rangle_1 \left|\tau_k\right\rangle \tag{3.85}$$

$$\text{With } \lambda(x,i,k) = \mathcal{A}(x) \wedge (i \oplus k) \vee (\neg\mathcal{A}(x) \wedge k) \tag{3.86}$$

So the eigenvalues of H_t are $\left|x\right\rangle H \cdot \left|i\right\rangle_1 \left|\tau_k\right\rangle$ with $x \in \{0,1\}^n, i \in \{0,1\}, k \in \{0,1\}$. The eigenvalues are $\lambda(x,i,k)$.

Because $\lambda(x,i,k)$ is a simple function on three Boolean values, it can be implemented on a small quantum (or even classical) circuit. The circuit W has the following action:

$$W\left(\left|x\right\rangle H \cdot \left|i\right\rangle_1 \left|\tau_k\right\rangle \left|0\right\rangle_1\right) = \left|x\right\rangle H \cdot \left|i\right\rangle_1 \left|\tau_k\right\rangle \left|1 \oplus \lambda(x,i,k)\right\rangle_1 \tag{3.87}$$

Perhaps we need some more convincing that λ can be implemented. After all, this term involves $\mathcal{A}$. Here we go, the circuit W is implemented as follows:

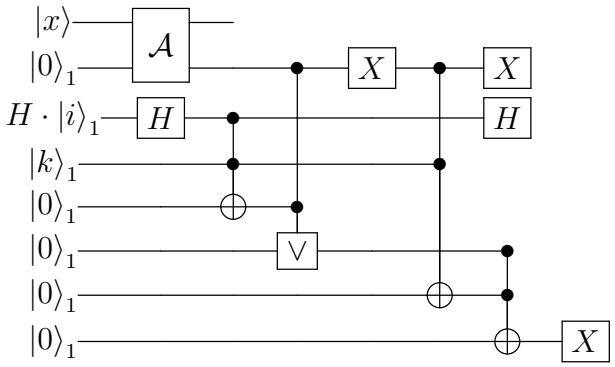

Figure 3.1: A circuit implementing W in Equation 3.87. It receives five ancilla qubits initialised to $\left|0\right\rangle_1$ as workspace.

The circuit W can be implemented, so the Hamiltonian can be evaluated, just as in the unrelativized case. This proves the theorem. □

Chapter 4

Acknowledgements

I woud like to thank Scott Aaronson, Harry Buhrman, Ralph Bottesch and Mikhail Vyalyi for many fruitful discussions, my supervisor André Deutz for his enthusiastic support of my work, and my friends for the things they put up with to try to understand what my research was about.

Bibliography

[1] Arora, Sanjeev, and Boaz Barak. Computational complexity: a modern approach. Cambridge University Press, 2009.

[2] Aharonov, Dorit, et al. "The pursuit for uniqueness: Extending Valiant-Vazirani theorem to the probabilistic and quantum settings." arXiv preprint arXiv:0810.4840 (2008).

[3] Meyer, Albert R., and Larry J. Stockmeyer. "The equivalence problem for regular expressions with squaring requires exponential space." SWAT (FOCS). 1972.

[4] Stockmeyer, Larry J. "The polynomial-time hierarchy." Theoretical Computer Science 3.1 (1976): 1-22.

[5] Goldreich, Oded. "Computational complexity: a conceptual perspective." ACM Sigact News 39.3 (2008): 35-39.

[6] Berman, Leonard, and Juris Hartmanis. "On isomorphisms and density of NP and other complete sets." SIAM Journal on Computing 6.2 (1977): 305-322.

[7] Aaronson, Scott, and Andrew Drucker. "A full characterization of quantum advice." Proceedings of the forty-second ACM symposium on Theory of computing. ACM, 2010.

[8] Agrawal, Manindra, Neeraj Kayal, and Nitin Saxena. "PRIMES is in P." Annals of mathematics (2004): 781-793.

[9] Marriott, Chris, and John Watrous. "Quantum arthur–merlin games." Computational Complexity 14.2 (2005): 122-152.

[10] Toda, Seinosuke. "PP is as hard as the polynomial-time hierarchy." SIAM Journal on Computing 20.5 (1991): 865-877.

[11] Babai, László, and Shlomo Moran. "Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes." Journal of Computer and System Sciences 36.2 (1988): 254-276.

[12] Sipser, Michael. "A complexity theoretic approach to randomness." Proceedings of the fifteenth annual ACM symposium on Theory of computing. ACM, 1983.

[13] Valiant, Leslie G., and Vijay V. Vazirani. "NP is as easy as detecting unique solutions." Theoretical Computer Science 47 (1986): 85-93.

[14] Valiant, Leslie G. "The complexity of computing the permanent." Theoretical computer science 8.2 (1979): 189-201.

[15] Grilo, Alex Bredariol, Iordanis Kerenidis, and Jamie Sikora. "QMA with subset state witnesses." International Symposium on Mathematical Foundations of Computer Science. Springer, Berlin, Heidelberg, 2015.

[16] Baker, Theodore, John Gill, and Robert Solovay. "Relativizations of the P=?NP question." SIAM Journal on computing 4.4 (1975): 431-442.

[17] Fortnow, Lance. "The role of relativization in complexity theory." Bulletin of the EATCS 52 (1994): 229-243.

[18] Hartmanis, Juris, and Richard E. Stearns. "On the computational complexity of algorithms." Transactions of the American Mathematical Society 117 (1965): 285-306.

[19] Gill, John. "Computational complexity of probabilistic Turing machines." SIAM Journal on Computing 6.4 (1977): 675-695.

[20] Kannan, Ravi. "Circuit-size lower bounds and non-reducibility to sparse sets." Information and Control 55.1-3 (1982): 40-56.

[21] Karp, Richard M., and Richard J. Lipton. "Some connections between nonuniform and uniform complexity classes." Proceedings of the twelfth annual ACM symposium on Theory of computing. ACM, 1980.

[22] Köbler, Johannes, Uwe Schöning, and Jacobo Torán. "Graph isomorphism is low for PP." Computational Complexity 2.4 (1992): 301-330.

[23] Adleman, Leonard M., Jonathan DeMarrais, and Ming-Deh A. Huang. "Quantum computability." SIAM Journal on Computing 26.5 (1997): 1524-1540.

[24] Kobayashi, Hirotada, François Le Gall, and Harumichi Nishimura. "Generalized quantum arthur-merlin games." Proceedings of the 30th Conference on Computational Complexity. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

[25] Aaronson, Scott. "Oracles are subtle but not malicious." 21st Annual IEEE Conference on Computational Complexity (CCC'06). IEEE, 2006.

[26] Aaronson, Scott, and Avi Wigderson. "Algebrization: A new barrier in complexity theory." ACM Transactions on Computation Theory (TOCT) 1.1 (2009): 2.

[27] Aaronson, Scott. "Quantum computing, postselection, and probabilistic polynomial-time." Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences. Vol. 461. No. 2063. The Royal Society, 2005.

[28] Aaronson, Scott. "On perfect completeness for QMA." arXiv preprint arXiv:0806.0450 (2008).

[29] Aaronson, Scott. "Quantum lower bound for recursive Fourier sampling." Quantum Information & Computation 3.2 (2003): 165-174.

[30] Aaronson, Scott. "Impossibility of succinct quantum proofs for collision-freeness." arXiv preprint arXiv:1101.0403 (2011).

[31] Aaronson, Scott. "A counterexample to the generalized Linial-Nisan conjecture." arXiv preprint arXiv:1110.6126 (2011).

[32] Shamir, Adi. "Ip= pspace." Journal of the ACM (JACM) 39.4 (1992): 869-877.

[33] Watrous, John. "PSPACE has constant-round quantum interactive proof systems." Foundations of Computer Science, 1999. 40th Annual Symposium on. IEEE, 1999.

[34] Drucker, Andrew, and Ronald de Wolf. "Quantum proofs for classical theorems." arXiv preprint arXiv:0910.3376 (2009).

[35] Cheung, Kevin KH, and Michele Mosca. "Decomposing finite abelian groups." arXiv preprint cs/0101004 (2001).

[36] Okamoto, Tatsuaki, and Keisuke Tanaka. Graph Non-Isomorphism Has a Succinct Quantum Certificate. Tokyo Institute of Technology. Department of Information Sciences, 2001.

[37] Aaronson, Scott, et al. "The space just above bqp." Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science. ACM, 2016.

[38] Yao, A. Chi-Chih. "Quantum circuit complexity." Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on. IEEE, 1993.

[39] Watrous, John. "Quantum algorithms for solvable groups." Proceedings of the thirty-third annual ACM symposium on Theory of computing. ACM, 2001.

[40] Hayden, Patrick, Kevin Milner, and Mark M. Wilde. "Two-message quantum interactive proofs and the quantum separability problem." Quantum Information & Computation 14.5&6 (2014): 384-416.

[41] Babai, László, Lance Fortnow, and Carsten Lund. "Non-deterministic exponential time has two-prover interactive protocols." Computational complexity 1.1 (1991): 3-40.

[42] Kobayashi, Hirotada, and Keiji Matsumoto. "Quantum multi-prover interactive proof systems with limited prior entanglement." Journal of Computer and System Sciences 66.3 (2003): 429-450.

[43]

[44] Fortnow, Lance, and John Rogers. "Complexity limitations on quantum computation." Computational Complexity, 1998. Proceedings. Thirteenth Annual IEEE Conference on. IEEE, 1998.

[45] Cubitt, Toby, and Ashley Montanaro. "Complexity classification of local Hamiltonian problems." SIAM Journal on Computing 45.2 (2016): 268-316.

[46] Childs, Andrew M., and Wim Van Dam. "Quantum algorithms for algebraic problems." Reviews of Modern Physics 82.1 (2010): 1.

[47] Bernstein, Ethan, and Umesh Vazirani. "Quantum complexity theory." SIAM Journal on Computing 26.5 (1997): 1411-1473.

[48] Bennett, Charles H., et al. "Strengths and weaknesses of quantum computing." SIAM journal on Computing 26.5 (1997): 1510-1523.

[49] Aaronson, Scott. "Quantum lower bound for the collision problem." Proceedings of the thiry-fourth annual ACM symposium on Theory of computing. ACM, 2002.

[50] Razborov, Alexander A., and Steven Rudich. "Natural proofs." Journal of Computer and System Sciences 55.1 (1997): 24-35.

[51] Aydinlioglu, Baris, and Eric Bach. "Affine Relativization: Unifying the Algebrization and Relativization Barriers." Electronic Colloquium on Computational Complexity (ECCC). Vol. 23. 2016.

[52] Fenner, Stephen, et al. "An oracle builder's toolkit." Information and Computation 182.2 (2003): 95-136.

[53] Watrous, John. "Quantum computational complexity." Encyclopedia of complexity and systems science. Springer New York, 2009. 7174-7201.

[54] Jain, Rahul, et al. "QIP=PSPACE." Communications of the ACM 53.12 (2010): 102-109.

[55] Beigi, Salman, Peter W. Shor, and John Watrous. "Quantum interactive proofs with short messages." arXiv preprint arXiv:1004.0411 (2010).

[56] Pereszlényi, Attila. "On quantum interactive proofs with short messages." arXiv preprint arXiv:1109.0964 (2011).

[57] Hayden, Patrick, Kevin Milner, and Mark M. Wilde. "Two-message quantum interactive proofs and the quantum separability problem." Quantum Information & Computation 14.5&6 (2014): 384-416.

[58] Vyalyi, Mikhail. "**QMA**=**PP** implies that **PP** contains **PH**." ECCCTR: Electronic Colloquium on Computational Complexity, technical reports. 2003.

[59] Vinodchandran, N. V. "A note on the circuit complexity of PP." Theoretical Computer Science 347.1-2 (2005): 415-418.

[60] Fortnow, Lance, Rahul Santhanam, and Ryan Williams. "Fixed-polynomial size circuit bounds." Computational Complexity, 2009. CCC'09. 24th Annual IEEE Conference on. IEEE, 2009.

[61] Buhrman, Harry, Lance Fortnow, and Thomas Thierauf. "Nonrelativizing separations." Computational Complexity, 1998. Proceedings. Thirteenth Annual IEEE Conference on. IEEE, 1998.

[62] Kitaev, Alexei Yu, Alexander Shen, and Mikhail N. Vyalyi. Classical and quantum computation. Vol. 47. Providence: American Mathematical Society, 2002.

[63] Aharonov, Dorit, and Tomer Naveh. "Quantum NP-a survey." arXiv preprint quant-ph/0210077 (2002).

[64] Kempe, Julia, and Oded Regev. "3-local Hamiltonian is QMA-complete." arXiv preprint quant-ph/0302079 (2003).

[65] Kempe, Julia, Alexei Kitaev, and Oded Regev. "The complexity of the local Hamiltonian problem." SIAM Journal on Computing 35.5 (2006): 1070-1097.

[66] Complexity Zoo. Retrieved from https://complexityzoo.uwaterloo.ca/Complexity_Zoo.