Master's thesis

On the Power of P with Access to a QMA Oracle

Dorian Rudolph

October 2020

Supervisor: Jun. Prof. Dr. Sevag Gharibian
2nd Examiner: Prof. Dr. Johannes Blömer

Contents

Abstract

The quantum analogue of NP, called QMA (Quantum Merlin Arthur) has the physically motivated complete problem of estimating the ground state energy of a local Hamiltonian. A related problem is simulating the measurement of a local Hamiltonian's ground state. Ambainis (CCC 2014) showed that this problem, denoted APX-SIM (Approximate Simulation), is $\mathrm{P}^{\mathrm{QMA[log]}}$-complete. $\mathrm{P}^{\mathrm{QMA[log]}}$ is the class of problems that can be solved by a deterministic polynomial-time Turing machine that may ask a QMA-oracle $O(\log(n))$ adaptive queries. Gharibian, Piddock, and Yirka (STACS 2020) show that a polynomial number of parallel queries can be simulated using a logarithmic number of adaptive queries and therefore $\mathrm{P}^{\mathrm{QMA[log]}} = \mathrm{P}^{\|\mathrm{QMA}}$. In the classical setting, an even stronger result is given by Gottlob (JACM 1995): A polynomial number of NP queries with a tree-like dependency graph can be simulated using a logarithmic number of adaptive queries (i.e., $\mathrm{P}^{\mathrm{NP[log]}} = \mathrm{TREES(NP)}$). We translate this result to the quantum setting and show $\mathrm{P}^{C[\log]} = \mathrm{TREES}(C)$ for $C \in \{\mathrm{MA}, \mathrm{QCMA}, \mathrm{QMA}, \mathrm{QMA}(2)\}$. We also improve the classical result by showing that even a query graph of bounded treewidth can be simulated with logarithmically many adaptive queries.

$\mathrm{P}^{\mathrm{QMA[log]}}$ can be viewed as an upper bound of QMA since equality seems unlikely. Another upper bound due to Vyalyi (ECCC 2013) is $\mathrm{A_0PP}$. We investigate whether one of these upper bounds is stronger than the other and give some evidence to the contrary by proving oracle separations between $\mathrm{coNP} \subseteq \mathrm{P}^{\mathrm{QMA[log]}}$ and $\mathrm{A_0PP}$, as well as between $\mathrm{SPP} \subseteq \mathrm{A_0PP}$ and $\mathrm{P}^{\|\mathrm{QMA}(2)} \supseteq \mathrm{P}^{\mathrm{QMA[log]}}$.

Gharibian and Sikora (TOCT 2018) introduce the ground state connectivity problem (GSCON), which asks whether there exists a sequence of m l-local unitaries that maps a ground state $|\psi\rangle$ to another ground state $|\phi\rangle$, such that intermediate states have energy at most $\lambda + \Delta$, where λ is the ground state energy. We investigate an exponential version, called $\mathrm{GSCON}_{\mathrm{exp}}$, with $l = 2$, inverse exponential Δ, and exponential m. We show that $\mathrm{GSCON}_{\mathrm{exp}}$ lies between PSPACE and NEXP. We define a quantum analogue of NPSPACE as potential upper bound for $\mathrm{GSCON}_{\mathrm{exp}}$, called QCMASPACE, with polynomially many qubits and an exponentially long classical proof. However, we are able to show QCMASPACE = NEXP. Finally, we prove that one can map any ground state to any other using $m \leq 2^{\mathrm{poly}(n)}$ 2-local unitaries, such that intermediate states have energy below $\lambda + \Delta$ for $\Delta = 2^{-\mathrm{poly}(n)}$.

Acknowledgements

I would like to thank Sevag Gharibian for many enlightening discussions and being a great supervisor. I also thank Rolando Diego Somma, William Kretschmer, and Stephen Piddock for helpful email exchanges.

Chapter 1

Introduction

We begin this thesis by motivating our topic of research and providing some background.

1.1 Motivation and Background

The complexity classes P and NP are central to theoretical computer science. The study of NP-completeness has helped us understand why so many problems appear to have no efficient algorithms. Cook and Levin have shown that any problem in NP can be reduced to the boolean satisfiability problem (3-SAT) in polynomial time [16, 48]. Karp has then shown that boolean satisfiability can in turn be reduced to a multitude of well-known combinatorial and graph theoretical problems [38]. Therefore, either none or all of these problems permit an efficient solution.

It is also of interest what lies beyond NP. Stockmeyer [57] defined the *polynomial hierarchy* (PH), of which the 0-th level is just P and subsequent levels are defined by giving a P, NP, or coNP machine oracle access to a class from the previous level. For example, we have $\Delta_2^{\mathrm{P}} = \mathrm{P}^{\mathrm{NP}}$ on the second level of PH. We can imagine this class as a Turing machine that may ask an *oracle* whether some boolean formula is satisfiable. P^{NP} has the following complete problem, as shown by Krentel [46]. Does the lexicographically largest satisfying assignment $x_1 \cdots x_r$ of a given satisfiable boolean formula have $x_r = 1$?

The next question one might ask is what happens if we impose restrictions on how the Turing machine in P^{NP} may ask its queries. We are particularly interested in $\mathrm{P}^{\mathrm{NP[log]}}$, where the Turing machine may only ask $O(\log n)$ queries on an input of length n. This class was first studied by Papadimitriou and Zachos [54]. As shown by Wagner [63, 62], the following problem is $\mathrm{P}^{\mathrm{NP[log]}}$-complete. Does the solution to a MAX-k-SAT instance have even Hamming weight (i.e., the assignment that maximizes the number of satisfied clauses)? Many similar problems involving the verification of a property of a solution to a polynomially bounded optimization problem are also in $\mathrm{P}^{\mathrm{NP[log]}}$ or even $\mathrm{P}^{\mathrm{NP[log]}}$-complete.

A different way to restrict P^{NP} is to say that all queries must be asked at the same time, or in other words, one query cannot depend on the outcome of another. This class is denoted by $\mathrm{P}^{\|\mathrm{NP}}$ and has been shown to be equal to $\mathrm{P}^{\mathrm{NP[log]}}$ by Hemachandra [35] as well as Buss and Hay [13].

Another way to view P^{NP} is as a directed acyclic graph (DAG) of queries with directed edges between two queries if one query depends on the result of another. By restricting the query dependency graph to a tree, this problem becomes $\mathrm{P}^{\mathrm{NP[log]}}$-complete, as shown by Gottlob [31].

These aforementioned results have analogues in the quantum setting. The research in that

direction began with Kitaev [43] in 1999, who proved in a "quantum Cook-Levin theorem", the problem of estimating the ground state energy of a k-local Hamiltonian for $k \geq 5$ is complete for the quantum analogue of NP, called Quantum Merlin Arthur (QMA). Formally, we are given a succinct description of a Hermitian matrix $H = \sum_i H_i \in \mathbb{C}^{2^n \times 2^n}$, where each H_i is a Hermitian that acts non-trivially on at most k qubits. The ground state is then the eigenvector of H with the smallest eigenvalue, which we call the ground state energy. This k-local Hamiltonian problem can be seen as a quantum analogue of the boolean satisfiability problem. We can think of the individual terms H_i as clauses. A quantum state $|\psi\rangle$ "satisfies" such a clause if it has low energy on H_i (i.e., $\langle\psi|H_i|\psi\rangle$ is small). This problem is physically motivated. The ground state corresponds to the state of a quantum system at low temperature. Since then, a multitude of other physical problems have been shown to be QMA-complete. We refer the reader to the surveys [53, 11, 23] and the references therein.

One such problem of particular interest is Approximate Simulation (APX-SIM), introduced by Ambainis [4]. Suppose we want to simulate the experiment of cooling down a quantum many-body system and then performing some measurement. Formally, we must decide, given Hamiltonian H describing the system and observable A describing the measurement, whether there exists a ground state $|\psi\rangle$ of H, such that $\langle\psi|A|\psi\rangle$ is below some threshold. Ambainis proved that this problem is $\mathrm{P}^{\mathrm{QMA[log]}}$-complete [4]. We remark the similarity to the classical $\mathrm{P}^{\mathrm{NP[log]}}$-complete problem of determining the Hamming weight of the assignment that satisfies the maximum number of clauses in a boolean formula.

This leads us to the question whether the results for parallel queries and query trees also hold in the quantum setting. Gharibian, Piddock, and Yirka [24] have shown that this is indeed the case (i.e., $\mathrm{P}^{\mathrm{QMA[log]}} = \mathrm{P}^{\|\mathrm{QMA}}$). They also prove the same result for NP and StoqMA by reductions to restricted variants of APX-SIM. This brings us to the first question we investigate in this thesis: Is $\mathrm{P}^{\mathrm{QMA}}$ with tree-like query dependencies also equal to $\mathrm{P}^{\mathrm{QMA[log]}}$? We answer this question in the affirmative and also show the same result for MA, QCMA, and QMA(2).

Another way to see $\mathrm{P}^{\mathrm{QMA[log]}}$ is as an upper bound to QMA. We believe that QMA is strictly smaller than $\mathrm{P}^{\mathrm{QMA[log]}}$ since it trivially holds that coQMA $\subseteq \mathrm{P}^{\mathrm{QMA[log]}}$, and coQMA $\subseteq$ QMA is generally not considered to be true. One might now ask for an upper bound to $\mathrm{P}^{\mathrm{QMA[log]}}$. It was shown by Beigel, Hemachandra, and Wechsung [9] that $\mathrm{P}^{\mathrm{NP[log]}} \subseteq$ PP, where PP is the class of problems solvable in probabilistic polynomial time with unbounded error (i.e., the acceptance probability difference between yes- and no-instances is positive but may be arbitrarily small). Gharibian and Yirka [26] prove that $\mathrm{P}^{\mathrm{QMA[log]}}$ is contained in PP. It further holds that QMA $\subseteq$ PP as noted by Kitaev and Watrous [45]. Vyalyi [61] gave a stronger result, namely that QMA $\subseteq \mathrm{A_0PP}$, and $\mathrm{A_0PP} = \mathrm{PP}$ would imply PH $\subseteq$ PP. A simple proof for QMA $\subseteq$ PP is also given by Marriott and Watrous [50]. So, we have two upper bounds on QMA: $\mathrm{P}^{\mathrm{QMA[log]}}$ and $\mathrm{A_0PP}$. One might ask whether one of these is contained in the other. We give some evidence to the contrary by providing oracle separations in both directions.

Returning to Hamiltonians, one might also be interested in properties of the ground space itself. Gharibian and Sikora [25] introduce the complexity class GSCON, which asks whether one ground state can be transformed into another using only local unitary gates, such that all intermediate states also have low energy. That research was partially inspired by the classical analogue: Given a 3-CNF formula ϕ and solutions x and y, does there exist a sequence of bitflips that transforms x into y, such that all intermediate strings are also valid solutions? Gopalan et al. [29] have shown that this problem is PSPACE-complete. GSCON permits many parameterizations. For a polynomially long

sequence of 2-local gates and at least inverse polynomial error terms, GSCON is QCMA-complete. For an exponentially long sequence of 1-local gates and inverse polynomial error terms, GSCON is PSPACE-complete. A succinct version of GSCON is even shown to be NEXP-complete. We ask now the question of how the complexity of the problem changes if exponentially many 2-local gates are allowed. We believe this GSCON parameterization, denoted $\text{GSCON}_{\text{exp}}$, to be of independent interest because its intermediate states do not necessarily have a succinct representation, which makes containment in PSPACE uncertain. We prove that $\text{GSCON}_{\text{exp}}$ lies between PSPACE and NEXP. Also, we investigate one idea for a better upper bound on $\text{GSCON}_{\text{exp}}$, namely a class with a polynomially space-bounded quantum verifier and an exponentially long classical proof. We call this problem QCMASPACE and show it to be equal to NEXP. Another observation we make is that in principle any pair of ground states of a Hamiltonian is connected through the span. We give an explicit construction for a sequence of 2-local unitaries that maps one state to another while keeping the distance to the span arbitrarily low. We conclude that $\text{GSCON}_{\text{exp}}$ with any error threshold becomes trivial when permitting sufficiently long unitary sequences.

1.2 Related Work

The previous section mainly served to motivate our line of research and illustrate how the investigated problems relate to each other. In the following, we give additional information on related research. Formal definitions of the relevant complexity classes can be found in Chapter 2.

$\text{P}^{\text{NP[log]}}$. As mentioned above, $\text{P}^{\text{NP[log]}}$ is contained in the second level of the polynomial hierarchy PH. It holds that $\text{P}^{\text{NP[log]}} = \text{P}^{\|\text{NP}}$ [35, 13]. Another well-known $\text{P}^{\text{NP[log]}}$-complete problem is determining a winner in Lewis Carroll's 1876 voting system, as proven by Hemaspaandra, Hemaspaandra, and Rothe [36]. Gottlob [31] showed that even P^{NP} with a tree-like query dependency graph equals $\text{P}^{\text{NP[log]}}$. We present a proof for that result in Section 3.1. The main idea is to construct a *total solution weight function* that assigns a weight to a given set of proofs and query results for a query graph. This formula is maximized by a correct solution. If it is polynomially bounded, the weight of the correct solution can be easily found using logarithmically many oracle queries. Gottlob gives an algorithm to reconfigure any tree into a suitable graph of low depth.

Other parameterizations $\text{P}^{\text{NP}[k]}$ have also been studied. It remains an open question to the best of our knowledge whether $\text{P}^{\text{NP}[k]}$, $\text{P}^{\text{NP}[\log^k n]}$ ($k = O(1)$), and P^{NP} coincide. One result due to Hartmanis [34] in that direction is that if $\text{P}^{\text{NP}[1]} = \text{P}^{\text{NP}[2]}$, then $\text{P}^{\text{NP}[1]} = \text{P}^{\text{NP[log]}}$ and PH collapses.

Hamiltonians. Kitaev's initial construction for a Hamiltonian from a QMA verifier is presented in Section 2.3 and uses a 5-local Hamiltonian. Kempe and Regev improved this to $k = 3$ [40]. Kempe, Kitaev, and Regev subsequently improved this to $k = 2$ [39]. Different parameterizations of the local Hamiltonian problems are also complete for different complexity classes. The *separable sparse Hamiltonian problem* due to Chailloux and Sattath [14] is QMA(2)-complete (QMA with two unentangled proofs) and asks if there exists a separable state (i.e., $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$) below a certain energy with respect to a given sparse Hamiltonian. Perhaps surprisingly, the separable *local* Hamiltonian problem is still contained in QMA. Fefferman and Lin [17] prove that QMA with an exponentially small gap (between the acceptance probabilities for yes- and no-instances) is PSPACE-complete, as well as the corresponding *precise local Hamiltonian problem*.

Ambainis [4] considers problems "slightly beyond QMA". Among these are estimating the spectral gap (i.e., the difference between the two smallest eigenvalues), which is contained in $\mathrm{P}^{\mathrm{QMA[log]}}$ and, most relevant to our line of research, the approximate simulation problem (APX-SIM): Given a k-local Hamiltonian H, and an l-local observable A, does there exist a ground state $|\psi\rangle$ of H such that $\langle\psi|A|\psi\rangle \leq \alpha$ (yes-case), or does it hold for all $|\psi\rangle$ with $\langle\psi|H|\psi\rangle \leq \lambda_{\min}(H) + \delta$ that $\langle\psi|A|\psi\rangle \geq \beta$ (no-case), where δ and $\beta - \alpha$ are lower-bounded by an inverse polynomial. $\lambda_{\min}(H)$ denotes the minimum eigenvalue (ground state energy) of H. APX-SIM is shown to be $\mathrm{P}^{\mathrm{QMA[log]}}$-complete for logarithmic k and l.

Gharibian and Yirka [26] improve this to $k = 5$ and $l = 1$. Gharibian, Piddock, and Yirka [24] further improve the result to $k = 2$ for physically motivated Hamiltonian models.

$\mathrm{P}^{\mathrm{QMA[log]}}$. As remarked by Gharibian and Yirka [26], an important difference between QMA and NP is the fact that QMA is a class of *promise problems*, whereas NP is a class of languages. Promise problems differ from languages in that they additionally have *invalid instances*. For example, in QMA a yes-instance is accepted with probability at least 2/3 and a no-instance with probability at most 1/3. But there may be strings whose acceptance probability lies strictly within that range. Those are invalid instances. As previously mentioned, decreasing that promise gap to an inverse exponential makes QMA equal to PSPACE. When giving a Turing machine oracle access to a promise class, it may ask invalid queries, which will be arbitrarily answered. We require that the Turing machine accepts yes-instances and rejects no-instances regardless of how invalid queries are answered. It may accept/reject arbitrarily on invalid instances.

To show that $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq$ APX-SIM, Ambainis constructs a "query Hamiltonian" (see Section 3.3.1) that contains the Hamiltonians for all possible queries the Turing machine may ask the oracle. Appropriate weights and terms that "activate" individual Hamiltonians depending on prior query answers ensure that the ground state contains a valid string of query answers. The result $\mathrm{P}^{C[\log]} = \mathrm{P}^{\|C}$ for $C \in \{\mathrm{NP}, \mathrm{StoqMA}, \mathrm{QMA}\}$ is shown using a similar query Hamiltonian construction [24]. The results for NP and StoqMA are obtained by restricting APX-SIM to certain classes of Hamiltonians and observables.

GSCON. The ground state connectivity problem is stated as follows [25]. Given a k-local Hamiltonian H and ground states $|\psi\rangle$ and $|\phi\rangle$ with $\langle\psi|H|\psi\rangle \leq \eta$ and $\langle\phi|H|\phi\rangle \leq \eta$, does there exist a sequence of l-local unitaries $U_1, \ldots, U_m$ such that $\|U_m \cdots U_1|\psi\rangle - |\phi\rangle\| \leq \varepsilon$ and $\langle\psi_i|H|\psi_i\rangle \leq \eta + \varepsilon$ for all intermediate states $|\psi_i\rangle = U_i \cdots U_1|\psi\rangle$? Besides the already mentioned classical motivation, GSCON is also physically motivated, namely with respect to quantum memories and stabilizer codes. One approach to implement quantum memory in a low temperature system is to encode the basis states $|\tilde{0}\rangle, |\tilde{1}\rangle$ in the ground space of a Hamiltonian with a sufficiently large spectral gap. The spectral gap ensures that noise does not easily take the system out of its ground space. But it also is necessary that noise does not alter the state in ground space, which is where GSCON comes into play. In a no-instance, no short unitary sequence (noise is generally assumed to be local) can map one ground state to another in low energy space. Therefore, we can be relatively certain that random noise will not change the stored state. This intuition is used in Kitaev's toy chain model [41] and the toric code [42, 44].

Note that as we show in this thesis, one can always construct exponentially long unitary sequences such that $\langle\psi_i|H|\psi_i\rangle \leq \eta + 2^{-\mathrm{poly}(n)}$ for all intermediate states. This does not appear to be an issue for the physical model since it seems extremely unlikely that these long and precise

sequences are caused by noise.

Gosset, Mehta, and Vidick [30] proved that GSCON is still QCMA-hard when restricted to commuting Hamiltonians (i.e., the local terms H_i pairwise commute).

A_0PP. The class A_0PP was introduced by Vyalyi [61] to prove that if QMA = PP, then PH is contained in PP. A problem is in A_0PP if there exists a GapP function g (i.e., the difference in the number of accepting and rejecting paths of a nondeterministic polynomial-time Turing machine) and a polynomial-time computable threshold function t, such that for a yes-instance x we have $g(x) \geq t(x)$ and $0 \leq g(x) \leq t(x)/2$ for a no-instance. Kuperberg [47] showed that $A_0PP = SBQP$, where SBQP is defined like BQP but with acceptance probability $\geq 2^{-p(n)}$ in the yes-case and $\leq 2^{-p(n)-1}$ in the no-case for some polynomial p.

1.3 Our Contribution

We give a brief overview of our results and proof techniques. Detailed definitions and proofs are given in the following chapters.

1.3.1 QMA Query Trees

The complexity class TREES(C) is defined as the set of promise problems that can be reduced to a tree of C-queries in polynomial time.

Theorem 1.1. $\text{TREES}(C) = \text{P}^{C[\log]}$ *for* $C \in \{\text{MA}, \text{QCMA}, \text{QMA}, \text{QMA}(2)\}$.

For C = QMA, we give two proofs. We begin by restating Gottlob's [31] proof for $\text{TREES(NP)} = \text{P}^{\text{NP}[\log]}$ (see Section 3.1), albeit slightly rephrased to better fit our needs. We model queries as circuits that receive as input the outputs of their dependencies, whereas Gottlob used boolean formulas with special linking variables that are set to 1 iff the associated formula is satisfiable. This change allows us to model the problem for different types of queries.

The first proof (see Section 3.3) is via reduction to APX-SIM. We construct a query Hamiltonian (similar to [4]) that contains Hamiltonians for each query. These Hamiltonians are constructed from QMA-verifier circuits using a modified version of Kitaev's Hamiltonian construction (see Section 2.3). The main difficulty is that the number of possible queries may be exponential. We work around this issue by having the query Hamiltonians operate on shared qubits, such that one Hamiltonian conceptually receives the result of the other as input. Individual Hamiltonian terms are weighted using a modified version of Gottlob's admissable weighting functions (see Definition 3.6). We also make use of his techniques to "flatten" the query tree (see Sections 3.1.2.3 and 3.3.3).

The second proof (see Section 3.4) takes a more general approach in that it adapts Gottlob's total solution weight function to the quantum setting (see (3.5) and (3.12)). This function receives as input a string of query results and accompanying proofs and outputs a sum of acceptance probabilities, weighted with an admissable weighting function. The main challenges in applying this approach to the quantum setting are dealing with invalid queries, and estimating the function in QMA.

The reasons for giving two proofs are that prior works used a reduction to APX-SIM [4, 26, 24] and that APX-SIM is a physically well motivated problem. On the other hand, it might seem quite indirect to go that route, especially for classical classes like MA. Therefore, we believe a direct

proof to be interesting in of itself. Furthermore, this proof is quite simple to generalize to other complexity classes, which we did for MA, QCMA, and QMA(2) (see Section 3.5).

1.3.2 Bounded Treewidth Query Graphs

We ask whether it possible to go beyond query trees. It seems natural to consider graphs of bounded treewidth, since they are "almost trees" in a certain sense. We call the complexity class of problems that can be reduced to an NP-query graph of bounded treewidth BTW(NP).

Theorem 1.2. $\mathrm{BTW(NP)} = \mathrm{P}^{\mathrm{NP[log]}}$.

We prove this result in Section 3.6 by constructing a tree of constantly sized separators for a given query graph. Then, we recursively decompose the graph by removing the separator at the root of the separator tree and creating copies of the entire disconnected graph for each possible combination of outputs from query nodes in the separator.

Unfortunately, we were not able to apply this approach to the quantum setting.

1.3.3 Upper Bounds for QMA

We compare the upper bounds $\mathrm{A_0PP}$ and $\mathrm{P}^{\mathrm{QMA[log]}}$ of NP. In attempting to show $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{A_0PP}$, we found a simple way to prove $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{PP}$, for which a rather involved proof is given in [26]. It is known that PP is closed under truth-table reductions (i.e., $\mathrm{P}^{\|\mathrm{PP}} = \mathrm{PP}$) [21]. Hence, we have $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{P}^{\|\mathrm{QMA}} \subseteq \mathrm{P}^{\|\mathrm{PP}} = \mathrm{PP}$ (see Theorem 4.1).

$\mathrm{A_0PP}$ does not seem to be closed under complement and therefore not under truth-table reductions. A problem is in $\mathrm{C_{=}P}$ if there exists a $g \in \mathrm{GapP}$ such that $g(x) = 0$ in the yes-case and $g(x) \neq 0$ in the no-case. $\mathrm{C_{=}P}$ is trivially contained in the complement of $\mathrm{A_0PP}$.

Theorem 1.3. $\mathrm{C_{=}P} \subseteq \mathrm{A_0PP}$ *implies* $\mathrm{PH} \subseteq \mathrm{PP}$.

The proof (see Section 4.1) is mainly a restatement of Vyalyi's [61] proof for "QMA = PP implies PH ⊆ PP".

We construct oracle separations between $\mathrm{A_0PP}$ and $\mathrm{P}^{\mathrm{QMA[log]}}$ to give some evidence that it is likely that neither one contains the other. In fact, we construct somewhat stronger separations.

Theorem 1.4. *There exists an oracle* A *such that* $\mathrm{coNP}^A \not\subseteq \mathrm{A_0PP}^A$.

We give a proof from first principles in Section 4.2.2. Therefor, we prove that if a polynomial of degree n is bounded at positions $1, \ldots, \Omega(1) \cdot n^2$, then it will also be small at 0. We then apply this observation to $\mathrm{A_0PP}^A$ and construct an oracle to separate coNP from $\mathrm{A_0PP}$ via diagonalization. We further prove that the containment $\mathrm{SBQP} \subseteq \mathrm{A_0PP}$ relativizes.

After devising our own proof, William Kretschmer pointed out a much simpler way to prove this result using query complexity, which we state in Section 4.2.3. One can view an oracle as a bitstring X of length $N = 2^n$ and show that an SBQP circuit requires $\Omega(\sqrt{N})$ queries to that bitstring to compute the AND-function on X, whereas a coNP verifier only needs a single query. Hence, our proof might not be of that much interest after all, since the techniques we used are quite similar to those employed in query complexity.

For the other direction, we prove the following separation in Section 4.3, where SPP is defined via a GapP function g such that $g(x) = 1$ in the yes-case and $g(x) = 0$ in the no-case, which is clearly contained in $\mathrm{A_0PP}$.

Theorem 1.5. *There exists an oracle A, such that* $\mathrm{SPP}^A \not\subseteq \mathrm{P}^{\|\mathrm{QMA}(2)^A}$

The main insight we use for the proof is that there must be strings x, such that for all queries $|\psi\rangle$ to the oracle, $\langle\psi|x\rangle$ is small. In other words, some strings will only be queried with exponentially small weight. Adding or removing these strings to A does not change the result of positively answered queries (the proof will still be accepted). By repeatedly adding or removing such strings to and from A, we can force any $\mathrm{P}^{\|\mathrm{QMA}(2)^A}$ verifier to make a mistake.

1.3.4 $\mathrm{GSCON}_{\mathrm{exp}}$

We define the class $\mathrm{GSCON}_{\mathrm{exp}}$ as GSCON with an exponentially small promise gap and exponentially long 2-local unitary sequences (see Definition 5.5). We are interested in how powerful that class is and show in Section 5.1

Theorem 1.6. $\mathrm{PSPACE} \subseteq \mathrm{GSCON}_{\mathrm{exp}} \subseteq \mathrm{NEXP}$.

Note that the containment in NEXP is trivial. To prove $\mathrm{PSPACE} \subseteq \mathrm{GSCON}_{\mathrm{exp}}$, we adapt the proof for $\mathrm{QCMA} \subseteq \mathrm{GSCON}$ due to Gharibian and Sikora [25]. Their proof works by constructing a Hamiltonian H from the QCMA verifier and embedding that Hamiltonian in a larger one. In the yes-case, the ground state is efficiently constructible and the starting state can be mapped to the target state via the ground state. In the no-case, it can be shown that some intermediate state must have significant overlap with H and will therefore have high energy. For $\mathrm{GSCON}_{\mathrm{exp}}$, we use the same approach but with the PSPACE-complete precise local Hamiltonian problem [17], using the fact that with exponentially many 2-local gates any state is constructible.

In an attempt to give a better upper bound than NEXP, we define a quantum analogue of NPSPACE, which we call QCMASPACE (see Section 5.2). It consists of an exponentially long quantum circuit with polynomially many qubits and an exponentially long classical proof. The proof is "streamed" to the verifier, which is modelled with special gates that are replaced with either X or I, depending on the associated proof bit. However, in a straightforward application of the PCP Theorem we show

Theorem 1.7. $\mathrm{QCMASPACE} = \mathrm{NEXP}$.

Another interesting question with regard to $\mathrm{GSCON}_{\mathrm{exp}}$ is which values make sense for the promise gap, specifically the bound on the energy of intermediate states. We argue in Section 5.3 that by making m sufficiently large, any $\mathrm{GSCON}_{\mathrm{exp}}$ instance becomes trivial.

Theorem 1.8. *Let $H \in \mathrm{Herm}(\mathbb{C}^d)$, $d = 2^n$ with $0 \preccurlyeq H \preccurlyeq I$, and $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$ such that $\langle\psi|H|\psi\rangle \leq \eta$ and $\langle\phi|H|\phi\rangle \leq \eta$. For $\varepsilon \geq 2^{-\mathrm{poly}(n)}$, we can construct a sequence of 2-local unitaries $U_1, \ldots, U_m$ with $m = O(2^{\mathrm{poly}(n)})$, such that*

$$\|U_m \cdots U_1|\psi\rangle - |\phi\rangle\|_\infty \leq \varepsilon,$$

and for all $|\psi_i\rangle := U_i \cdots U_1|\psi\rangle$

$$\langle\psi_i|H|\psi_i\rangle \leq \eta + \varepsilon.$$

For this proof, we devise a general decomposition of unitaries close to identity into 2-local gates close to identity. The prior 2-local decompositions known to us make use of gates far from identity (e.g., CNOT). That seems excessive to decompose a unitary $U = e^{itH}$ with very small t. We give an

approximate decomposition $e^{itH} \approx \prod_j e^{it_j H_j}$, where $\sum_j |t_j|$ is bounded by $2^{\mathrm{poly}(n)}|t|^{1/2n}$. Basically, we can decompose a unitary with a short *pulse time* into many local unitaries with short pulse times. For that, we first write $H = \sum_j \alpha_j P_j$ in the Pauli basis (i.e., each P_j is a tensor product of the Pauli matrices and identity) and apply the Suzuki decomposition [58] (see Lemma 5.14)

$$e^{iH} = \prod_j e^{i\alpha_j P_j} + O(t^2),$$

where $\sum_j |\alpha_j| \leq t$. Clinton, Bausch, and Cubitt [15] give an exact 2-local decomposition for the $e^{i\alpha_j P_j}$ terms with bounded pulse times. We use a slightly different decomposition since we were not able to verify the proof sketch given in [15].

1.4 Future Work

We identify several open problems that we find worth researching further. We reduce TREES(QMA) to APX-SIM with a 6-local Hamiltonian in Section 3.3. Can this be improved (e.g., using [39])?

For which classes C does $\mathrm{TREES}(C) = \mathrm{P}^{C[\log]}$ also hold? Other quantum classes like StoqMA and UQMA seem like appropriate candidates. We discuss StoqMA briefly in Section 3.5. UQMA (unique QMA) is defined by Aharonov et al. [2] similar to QMA but with the restriction that in the yes-case, there exists a proof $|\psi\rangle$ that is accepted with probability at least c and proofs orthogonal to $|\psi\rangle$ are accepted with probability at most $s < c - 1/\operatorname{poly}$.

We would also like to extend the results for bounded treewidth (see Section 3.6) to promise problems and thereby QMA and its variants, as we did for $\mathrm{TREES}(C)$.

It would also be interesting to find a better upper bound for QMA than $\mathrm{A_0PP}$ and $\mathrm{P}^{\mathrm{QMA}[\log]}$. Their intersection might be a possible candidate, although it is not clear to us which problems besides QMA are contained in $\mathrm{A_0PP} \cap \mathrm{P}^{\mathrm{QMA}[\log]}$. It seems hard to utilize the multiple queries from $\mathrm{P}^{\mathrm{QMA}[\log]}$ due to the one-sidedness of $\mathrm{A_0PP}$.

Moreover, we would like to strengthen the oracle separation of SPP and $\mathrm{P}^{\|\mathrm{QMA}(2)}$ to adaptive queries (i.e., $\mathrm{P}^{\mathrm{QMA}(2)}$).

For $\mathrm{GSCON}_{\mathrm{exp}}$, we would like to find tighter bounds than PSPACE and NEXP. It seems $\mathrm{GSCON}_{\mathrm{exp}}$ should be harder than PSPACE as intermediate states do not necessarily have a succinct representation, but we were not able to show NEXP-completeness.

We have proven that $\mathrm{GSCON}_{\mathrm{exp}}$ becomes trivial for sufficiently large m. Does the same also hold for the 1-local case (see Section 5.3.4.1), or does it remain PSPACE-hard (or even complete) for unbounded m?

The decomposition given in Section 5.3.3 is inexact due to the use of the Suzuki decomposition. Is there an exact decomposition with similar properties regarding pulse times?

Chapter 2

Foundations

In this chapter, we introduce notation that is used throughout the thesis (see Section 2.1). We further define the complexity classes from prior work that we make use of (see Section 2.2). Afterwards, we restate the well known Hamiltonian construction used by Kitaev [43] to prove that the k-local Hamiltonian problem is QMA-complete (see Section 2.3).

2.1 Notation and Useful Results

In this section, we define the notation used throughout this thesis and give a collection of well-known definitions and results we use.

Let $\mathbb{N} = \{0, 1, 2, \dots\}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ be the sets of natural, integer, rational, real, and complex numbers, respectively. We denote the real part of a complex number c by $\mathrm{Re}(c)$. We define $[n] := \{1, \dots, n\}$. Let S be a set. We define the *characteristic function* χ_S, such that $\chi_S(x) = 1$ if $x \in S$ and $\chi_S(x) = 0$ otherwise. The *superset* of S is denoted by $2^S := \{S' \mid S' \subseteq S\}$. We write $S_1 \uplus S_2$ for the union of disjoint sets.

The *degree* of a polynomial p is denoted $\deg(p)$. For $n \in N$, we define the *falling factorial* $(x)_n := \prod_{k=1}^{n}(n - k + 1)$.

$\{0,1\}^*$ denotes the set of all binary strings. For $x, y \in \{0,1\}^*$, $x \circ y$ denotes the concatenation of x and y.

We write $\mathrm{poly}(n)$ as a placeholder for an arbitrary polynomial. For example, we say $f(n) \leq \mathrm{poly}(n)$, if $f(n) = O(n^c)$ for some $c > 0$. Similarly, we say $f(n) \geq 2^{-\mathrm{poly}(n)}$, if $f(n) = \Omega(2^{-O(n^c)})$ for some $c > 0$.

2.1.1 Probability Theory

For a random event A, we denote the probability that A occurs by $\Pr[A]$. For a random variable X, we denote its expected value by $\mathbb{E}[X]$.

Lemma 2.1 (Hoeffding's inequality [37])**.** *Let $X_1, \dots, X_n$ be independent random variables, such that each X_i is within the real interval $[a_i, b_i]$. Their empirical mean is defined as*

$$\overline{X} = \frac{1}{n}(X_1 + \cdots + X_n).$$

For $t > 0$,

$$\Pr\left[\left|\overline{X} - \mathbb{E}[\overline{X}]\right| \geq t\right] \leq 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$

2.1.2 Graph Theory

Let $G = (V, E)$ be a directed graph. For a node $v \in V$, we define $\mathrm{indeg}(v)$ and $\mathrm{outdeg}(v)$ as the number of incoming and outgoing edges, respectively. $\mathrm{pred}(v) := \{w \in V \mid (w, v) \in E\}$ denotes the set of immediate predecessors of v and $\mathrm{suc}(v) := \{w \in V \mid (v, w) \in E\}$ the set of immediate successors of v. If G contains no directed cycles, we call it a DAG (directed acyclic graph).

Definition 2.2 (Tree decomposition)**.** Let $G = (V, E)$ be an undirected graph. A *tree decomposition* T of G is a graph with nodes $X_1, \ldots, X_m \subseteq V$ satisfying the following properties:

- Each node of G is contained in a node of T: $\bigcup_{i=1}^{m} X_i = V$.
- For all $v \in V$, the nodes $X_i \ni v$ form a connected subtree of T.
- For all $e \in E$, there exists an X_i such that $e \subseteq X_i$.

The *width* of a tree decomposition T is defined as $\mathrm{width}(T) := \max_i |X_i| - 1$. The *treewidth* of G, denoted $\mathrm{tw}(G)$, is defined as the minium width among all possible tree decompositions of G.

Bodlaender has shown that tree decompositions for graphs with bounded treewidth (i.e., $\mathrm{tw}(G) = O(1)$) can be computed in linear time [10].

Definition 2.3 (Separator number [32])**.** Let $G = (V, E)$ be an undirected graph. A set $S \subseteq V$ is a *separator* of G if $G \setminus S$ (i.e., the graph induced by the nodes $V \setminus S$) has at at least two connected components or at most one node. S is *balanced* if every connected component of $G \setminus S$ has at most $\lceil (|V| - |S|)/2 \rceil$ nodes.

The *balanced separator number* of G, denoted $\mathrm{s}(G)$, is defined as the smallest k such that for every $Q \subseteq G$, the induced subgraph $G[Q]$ has a balanced separator of size at most k.

Lemma 2.4 ([32])**.** $\mathrm{s}(G) \leq \mathrm{tw}(G)$.

We define tree decompositions and separator number for a directed graph simply as that of its undirected version.

2.1.3 Quantum Computation

Generally, we refer to Nielsen and Chuang [51] for an introduction to quantum computation. This subsection rather serves to refresh the reader's memory and to establish notation.

Matrices. The set of linear operators on a vector space V is denoted by $\mathcal{L}(V)$ (e.g., $\mathcal{L}(\mathbb{C}^d)$ consists of the $d \times d$ matrices). We define the *Hermitian conjugate* of a matrix $A \in \mathcal{L}(\mathbb{C}^d)$ as the complex conjugate of its transpose, denoted by $A^\dagger := \overline{A^T}$. A is *Hermitian* if $A^\dagger = A$. The set of Hermitian matrices is $\mathrm{Herm}(\mathbb{C}^d)$. A is *unitary* if $A^\dagger A = AA^\dagger = I$. The set of unitaries is $\mathcal{U}(\mathbb{C}^d)$. We denote the *trace* of a matrix $A \in \mathcal{L}(\mathbb{C}^d)$ by $\mathrm{Tr}(A)$. The *commutator* of two matrices A and B is given by $[A, B] := AB - BA$. We say A and B *commute* if $[A, B] = 0$ and *anti-commute* if $AB + BA = 0$.

Hermitian and unitary matrices are *normal* (i.e., $A^\dagger A = AA^\dagger$). Therefore, they have a spectral decomposition

$$A = \sum_{j=1}^{d} \lambda_i |\lambda_j\rangle\langle\lambda_j|,$$

where $\{|\lambda_j\rangle\}$ is an orthonormal basis of $\mathbb{C}^d$ and $|\lambda_j\rangle$ is an eigenvector with eigenvalue for λ_j. For unitary matrices, we have $\lambda_j = e^{i\theta_j}$ for some $\theta_j \in \mathbb{R}$. For Hermitian matrices, we have $\lambda_j \in \mathbb{R}$. We say a Hermitian matrix H is positive semi-definite if all of its eigenvalues are nonnegative. We write $A \succcurlyeq B$ if $A - B$ is positive semi-definite, and $A \succ B$ if $A - B$ is positive definite.

Pauli matrices. The Pauli matrices are given by

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

$\mathrm{Herm}(\mathbb{C}^2)$ can be seen as a 4-dimensional real vector space with the Frobenius inner product $\langle A, B\rangle = \mathrm{Tr}(A^\dagger B)$. Note that the norm given by $\sqrt{\langle A, A\rangle}$ is equivalent to the Frobenius norm. The *Pauli basis* $\{I, X, Y, Z\}$ is an orthogonal basis of $\mathrm{Herm}(\mathbb{C}^2)$. This can easily be verified by observing that all pairs of matrices have an inner product of 0 and the dimension $\mathrm{Herm}(\mathbb{C}^2)$ is bounded by 4. We can also view these matrices as 4-dimensional vectors. The Frobenius inner product is then equivalent to the usual inner product on the vectors.

The Pauli basis generalizes to higher dimensions, such that $\mathrm{Herm}((\mathbb{C}^2)^{\otimes n})$ is a 2^{2n}-dimensional real vector space with orthogonal basis

$$\{I, X, Y, Z\}^{\otimes n} = \big\{P_1 \otimes \cdots \otimes P_n \mid P_1, \ldots, P_n \in \{I, X, Y, Z\}\big\}.$$

Correctness follows again from dimension and pairwise orthogonality.

Operator functions. For a function $f : \mathbb{C} \to \mathbb{C}$, we define the *operator function* on a Hermitian or unitary matrix by applying f to its eigenvalues:

$$f(A) = \sum_{j=1}^{d} f(\lambda_i)|\lambda_j\rangle\langle\lambda_j|$$

Hence, for $U \in \mathcal{U}(\mathbb{C}^d)$, we have

$$U = \sum_{j=1}^{d} e^{i\theta_j}|\lambda_j\rangle\langle\lambda_j| = e^{iH}, \qquad H = \sum_{j=1}^{d} \theta_j|\lambda_j\rangle\langle\lambda_j|.$$

Quantum states. We use the bra-ket notation for quantum states, where $|\psi\rangle$ indicates a column vector and $\langle\psi| = |\psi\rangle^\dagger$ a row vector. We generally assume that any such vector (e.g., $|\psi\rangle$) is a unit vector (i.e., $\langle\psi|\psi\rangle = 1$). The standard basis of $\mathbb{C}^d$ is usually written as $\{|j\rangle\}$. The vector space associated with a single qubit is $\mathcal{B} := \mathbb{C}^2$. n qubits have the vector space $\mathcal{B}^{\otimes n} = \mathcal{B}_1 \otimes \cdots \otimes \mathcal{B}_n$, where $\otimes$ is the tensor product. We identify individual subspaces of the tensor product (also called *registers*) by subscripts. For example, $|0\rangle_B|1\rangle_A = |1\rangle_A \otimes |0\rangle_B \in \mathcal{B}_A \otimes \mathcal{B}_B$. This allows us to reorder notation conveniently without changing its semantics.

We also represent quantum states as density *density operators*. A matrix $\rho \in \mathrm{Herm}(\mathbb{C}^d)$ is a density operator if $\rho \succcurlyeq 0$ and $\mathrm{Tr}(\rho) = 1$. The density operator of a *pure state* $|\psi\rangle \in \mathbb{C}^d$, is $\rho = |\psi\rangle\langle\psi|$. A density operator of rank > 1 is a mixed state. It essentially describes a distribution over pure states.

Measurement. A matrix Π is a projector if $\Pi^2 = \Pi$ (i.e., Π has only eigenvalues 0 and 1). A projective measurement is described by a set of projectors $\{\Pi_i\}_{i=1}^m$ such that $\sum_{i=1}^m \Pi_i = I$. Measurement in the standard basis is given by $\{|i\rangle\langle i|\}$. The probability of obtaining outcome i when measuring state ρ is given by $\mathrm{Tr}(\Pi_i \rho)$. The state after measuring outcome i is given by $\rho' = \Pi_i \rho \Pi_i / \mathrm{Tr}(\Pi_i \rho)$.

A more general notion of measurement is the *POVM formalism* (Positive Operator-Valued Measure). A POVM is defined by a set of operators $\{E_i\}_{i=1}^m$ with $E_i \succcurlyeq 0$ and $\sum_{i=1}^m E_i = I$. The probability of observing outcome i is still given by $\mathrm{Tr}(E_i \rho)$. POVMs can be simulated by projective measurements in a larger space [43].

Norms. For vectors $v \in \mathbb{C}^d$, we define the p-norm for $p \geq 1$

$$\|v\|_p := \left(\sum_{i=1}^d |v_i|^p \right)^{1/p}.$$

We call $\|\cdot\|_2$ also the *Euclidean norm*.

Lemma 2.5 ([28, Equation 2.2.5])**.** *For all $v \in \mathbb{C}^d$,*

$$\|v\|_2 \leq \|v\|_1 \leq \sqrt{d}\|v\|_2.$$

Lemma 2.6 ([22])**.** *For all $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$,*

$$\||\psi\rangle - |\phi\rangle\|_2 = \sqrt{2 - 2\,\mathrm{Re}(\langle\phi|\psi\rangle)}.$$

Proof.

$$\begin{aligned}
\||\psi\rangle - |\phi\rangle\|_2 &= \sqrt{(\langle\phi| - \langle\phi|)(|\psi\rangle - \langle\phi|)} \\
&= \sqrt{\langle\psi|\psi\rangle - \langle\psi|\phi\rangle - \langle\phi|\psi\rangle + \langle\phi|\phi\rangle} \\
&= \sqrt{2 - 2\,\mathrm{Re}(\langle\psi|\phi\rangle)}
\end{aligned}$$

□

For operators $M \in \mathcal{L}(\mathbb{C}^d)$, we define the corresponding *operator norm*, usually called the *spectral norm*, as

$$\|M\|_\infty := \max_{v \in \mathbb{C}^d} \frac{\|Mv\|_2}{\|v\|_2}.$$

It holds that $\|M\|_\infty = \sqrt{\lambda_{\max}(M^\dagger M)}$. For $M \succcurlyeq 0$, we have $\|M\|_\infty = \lambda_{\max}(M)$, where $\lambda_{\max}(M)$ denotes the largest eigenvalue of M. We write $M = O(f(d))$ if $\|M\|_\infty = O(f(d))$ for some function f.

The *Frobenius norm* is defined as

$$\|M\|_{\mathrm{F}} := \sqrt{\sum_{i=1}^d \sum_{j=1}^d |m_{ij}|^2},$$

where m_{ij} denote the entries of M. Note that the Frobenius norm is the same as the Euclidean norm of M viewed as a d^2-dimensional vector.

Lemma 2.7 ([28, Equation 2.3.7])**.**

$$\|M\|_\infty \leq \|M\|_{\mathrm{F}} \leq \sqrt{d}\|M\|_\infty$$

We define the *trace norm* for operators $M \in \mathcal{L}(\mathbb{C}^d)$ as

$$\|M\|_{\mathrm{tr}} := \mathrm{Tr}(|M|) = \mathrm{Tr}\sqrt{M^\dagger M},$$

where $|\cdot|$ and $\sqrt{\cdot}$ are applied as operator functions.

Lemma 2.8 ([22])**.** *Let $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$. Then,*

$$\||\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|\|_{\mathrm{tr}} = 2\sqrt{1 - |\langle\psi|\phi\rangle|^2}. \tag{2.1}$$

Proof. Let $M = |\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|$. M has rank ≤ 2 and $\mathrm{Tr}(M) = 0$. Hence, $M = \lambda|\lambda_1\rangle\langle\lambda_1| - \lambda|\lambda_2\rangle\langle\lambda_2|$ for some $\lambda \geq 0$. Therefore, $2\lambda^2 = \mathrm{Tr}(M^2) = 2 - 2|\langle\psi|\phi\rangle|^2$. Thus, $\lambda = \sqrt{1 - |\langle\psi|\phi\rangle|^2}$. (2.1) follows due to $\|M\|_{\mathrm{tr}} = 2\lambda$. □

2.2 Complexity Classes

Due to the inherently probabilistic nature of quantum computation, the quantum complexity classes we are interested in are defined in terms of *promise problems*. A promise problem Π is defined by a tuple $\Pi = (\Pi_{\mathrm{yes}}, \Pi_{\mathrm{no}}, \Pi_{\mathrm{inv}})$ with $\Pi_{\mathrm{yes}} \cup \Pi_{\mathrm{no}} \cup \Pi_{\mathrm{inv}} = \{0,1\}^*$. We call $x \in \Pi_{\mathrm{yes}}$ a *yes-instance*, $x \in \Pi_{\mathrm{no}}$ a *no-instance*, and $x \in \Pi_{\mathrm{inv}}$ an *invalid instance*. Having invalid instances is what distinguishes promise problems from languages.

2.2.1 Quantum Complexity Classes

The complexity class BQP can be considered as the "quantum analouge" of P. The circuits used by BQP and other quantum complexity classes belong to *polynomial-time uniform quantum circuit families* $\{Q_n\}$. That means, there exists a Turing machine that on input n outputs a classical description of a quantum circuit Q_n in time $\mathrm{poly}(n)$.

Definition 2.9 (BQP)**.** Fix a polynomial $q(n)$. A promise problem Π is in BQP (Bounded-Error Quantum Polynomial-Time) if there exists a polynomial-time uniform quantum circuit family $\{Q_n\}$, such that the following holds:

- For all n, $Q_n \in \mathcal{U}\left(\mathcal{B}_A^{\otimes n} \otimes \mathcal{B}_C^{\otimes q(n)}\right)$. The register A is used for the input and C contains ancillae initialized to $|0\rangle$.
- $\forall x \in \Pi_{\mathrm{yes}} : \Pr[Q_{|x|} \text{ accepts } |x\rangle] \geq 2/3$
- $\forall x \in \Pi_{\mathrm{no}} : \Pr[Q_{|x|} \text{ accepts } |x\rangle] \leq 1/3$

To check if a circuit Q_n accepts the input, we measure the first qubit of register C in standard basis. If it is $|1\rangle$, we say the circuit accepts the input. For $x \in \{0,1\}^n$, the probability that Q_n accepts x is given by

$$\Pr[Q_n \text{ accepts } |x\rangle] = \left\|\left(I_A \otimes |1\rangle\langle 1|_{C_1} \otimes I_{C_{2\ldots q(n)}}\right) Q_n |x\rangle_A |0\rangle_C\right\|_2^2.$$

If BQP is the quantum analogue of P, then what is the quantum analogue of NP? The arguably

most natural analogue is QMA, where a BQP-verifier is given an additional quantum proof. As the name suggests, QMA can also be seen as a quantum analogue of MA.

Definition 2.10 (QMA)**.** Fix polynomials $p(n)$ and $q(n)$. A promise problem Π is in QMA (Quantum Merlin Arthur) if there exists a polynomial-time uniform quantum circuit family $\{Q_n\}$ such that the following holds:

- For all n, $Q_n \in \mathcal{U}\left(\mathcal{B}_A^{\otimes n} \otimes \mathcal{B}_B^{\otimes p(n)} \otimes \mathcal{B}_C^{\otimes q(n)}\right)$. The register A is used for the input, B contains the proof, and C the ancillae initialized to $|0\rangle$.
- $\forall x \in \Pi_{\text{yes}}\ \exists |\psi\rangle \in \mathcal{B}^{\otimes p(|x|)} : \Pr[Q_{|x|} \text{ accepts } |x\rangle|\psi\rangle] \geq 2/3$
- $\forall x \in \Pi_{\text{no}}\ \forall |\psi\rangle \in \mathcal{B}^{\otimes p(|x|)} : \Pr[Q_{|x|} \text{ accepts } |x\rangle|\psi\rangle] \leq 1/3$

If we modify QMA to require a classical proof $|y\rangle$ instead of $|\psi\rangle$, we get the complexity class QCMA.

Note that the thresholds $c = 2/3$ and and $s = 1/3$ may be replaced with $c = 1 - \varepsilon$ and $s = \varepsilon$ such that $\varepsilon \geq 2^{-\operatorname{poly}(n)}$ [43]. We also refer to c as *completeness*, s as *soundness*, and $c - s$ as the *promise gap*.

If we modify QMA to receive k unentangled proofs (i.e., $|\psi\rangle = \bigotimes_{j=1}^{k} |\psi_j\rangle$), we get the complexity class QMA(k). It holds that QMA(2) = QMA(poly(n)) as shown by Harrow and Montanaro [33]. Therefore, probability amplification is possible.

In this thesis, we examine some complexity classes where a Turing machine is given access to an oracle for another complexity class.

Definition 2.11 (P^C)**.** Let C be a complexity class with complete problem Π. $\mathrm{P}^C = \mathrm{P}^\Pi$ is the class of (promise) problems that can be decided by a polynomial-time deterministic Turing machine M with the ability to query an oracle for Π. If M asks an *invalid* query $x \in \Pi_{\text{inv}}$, the oracle may respond arbitrarily.

We say $\Gamma \in \mathrm{P}^C$ if there exists an M as above such that M accepts/rejects for $x \in \Gamma_{\text{yes}}/x \in \Gamma_{\text{no}}$, regardless of how invalid queries are answered.

For a function f, we define $\mathrm{P}^{C[f]}$ in the same way, but with the restriction that M may ask at most $O(f(n))$ queries on input of length n.

For an integer k, we define $\mathrm{P}^{C[k]}$, where M may ask at most k queries on each input.

$\mathrm{P}^{\|C}$ denotes the class where M must ask all queries at the same time. We call these queries *non-adaptive* opposed to the *adaptive* queries of the above classes, because the queries do not depend on the results of other queries.

For a function $f : \{0,1\}^* \to \{0,1\}^*$, we define P^f and the other classes analogously, except that M may now query the oracle for values $f(x)$.

The $\mathrm{P}^{\mathrm{QMA[log]}}$-complete problem is APX-SIM. It essentially asks whether a given Hamiltonian has a ground state with a certain property (e.g., a ground state where the first qubit is set to $|1\rangle$).

Definition 2.12 (APX-SIM [4])**.** Fix a polynomial $p(n)$. The promise problem APX-SIM (Approximate Simulation) is defined as follows.

Input:

- A k-local Hamiltonian $H \in \operatorname{Herm}(\mathcal{B}^{\otimes n})$ (see Definition 2.28).
- A k-local observable $A \in \operatorname{Herm}(\mathcal{B}^{\otimes n})$.
- Efficiently computable threshold functions $\alpha, \beta, \delta : \mathbb{N} \to \mathbb{R}$ satisfying the *promise gap* $\beta(n) - \alpha(n) \geq 1/p(n)$ and $\delta(n) \geq 1/p(n)$ for all $n \geq 1$.

Output:

YES: H has a ground state $|\psi\rangle$ satisfying $\langle\psi|A|\psi\rangle \leq \alpha(n)$.
NO: For all $|\psi\rangle \in \mathcal{B}^{\otimes n}$ with $\langle\psi|H|\psi\rangle \leq \lambda_{\min}(H) + \delta$, it holds that $\langle\psi|A|\psi\rangle \geq \beta(n)$.

Ambainis shows completeness for $k = \Theta(\log n)$ [4]. Gharibian and Yirka [26] improve this to $k = 5$ by using an approach similar to Section 2.3.2.4 and observables that measure only a single qubit. Gharibian, Piddock, and Yirka [24] improved this to $k = 2$ for physically motivated Hamiltonian models.

Definition 2.13 (StoqMA [12])**.** Fix polynomials $\alpha(n), \beta(n), p(n), q(n), r(n)$ with $\alpha(n) - \beta(n) \geq 1/\operatorname{poly}(n)$. A promise problem Π is in StoqMA (Stoquastic Merlin Arthur) if there exists a polynomial-time uniform quantum circuit family $\{Q_n\}$ such that the following holds:

- For all n, $Q_n \in \mathcal{U}\left(\mathcal{B}_A^{\otimes n} \otimes \mathcal{B}_B^{\otimes p(n)} \otimes \mathcal{B}_C^{\otimes q(n)} \otimes \mathcal{B}_D^{\otimes r(n)}\right)$. The register A is used for the input, B contains the proof, C ancillae initialized to $|0\rangle$, and D ancillae initialized to $|+\rangle$. Q_n only uses X, CNOT, and Toffoli gates.
- For $x \in \{0,1\}^*$, $|x| = n$, $|\psi\rangle \in \mathcal{B}^{\otimes p(n)}$, let $|\psi_{\text{in}}\rangle := |x\rangle_A |\psi\rangle_B |0\rangle_C^{\otimes q(n)} |+\rangle_D^{\otimes r(n)}$. The acceptance probability is then given by

$$\Pr[Q_n \text{ accepts } |x\rangle|\psi\rangle] = \langle\psi_{\text{in}}| Q_n^\dagger \Pi_{\text{acc}} Q_n |\psi_{\text{in}}\rangle,$$

  where $\Pi_{\text{acc}} = |+\rangle\langle+|_{C_1}$ measures the first ancilla in the $\{|+\rangle, |-\rangle\}$ basis.
- $\forall x \in \Pi_{\text{yes}}\ \exists |\psi\rangle \in \mathcal{B}^{\otimes p(|x|)} : \Pr[Q_{|x|} \text{ accepts } |x\rangle|\psi\rangle] \geq \alpha(n)$
- $\forall x \in \Pi_{\text{no}}\ \forall |\psi\rangle \in \mathcal{B}^{\otimes p(|x|)} : \Pr[Q_{|x|} \text{ accepts } |x\rangle|\psi\rangle] \leq \beta(n)$

Note that the only difference between StoqMA and MA is that StoqMA may perform its final measurement in the $\{|+\rangle, |-\rangle\}$ basis (i.e., setting $\Pi_{\text{acc}} = |0\rangle\langle 0|_{C_1}$ would result in MA) [12].

It further holds that the StoqMA verifier accepts any state with only nonnegative coordinates with probability $\geq 1/2$. Therefore, we cannot amplify the gap by majority voting as for MA. Recently, Aharonov, Grilo, and Liu [3] have shown that StoqMA with $\alpha(n) = 1 - \text{negl}(n)$ and $\beta(n) = 1 - 1/\operatorname{poly}(n)$ is contained in MA, where $\text{negl}(n)$ denotes a function smaller than all inverse polynomials for sufficiently large n. It is therefore unlikely that such an amplification is possible.

2.2.2 Counting Complexity Classes

Next, we introduce some definitions needed for studying the upper bounds of QMA.

Definition 2.14 (FP)**.** FP is the class of functions $f : \{0,1\}^* \to \mathbb{Z}$ that can be computed by a polynomial-time Turing machine.

Let M be a polynomial-time nondeterministic Turing machine with two halting states: *accept* (1) and *reject* (0). We call such a Turing machine a *counting machine* (CM). We denote the number of accepting computation paths on input x by $\#M(x)$.

Definition 2.15 (#P [60])**.** $\#\text{P} = \{\#M \mid M \text{ is a CM}\}$.

Definition 2.16 (GapP [20])**.** Denote by $\overline{M}$ the CM that acts like M but accepts whenever M rejects and rejects whenever M accepts. We define the function

$$\text{gap}_M : \{0,1\}^* \to \mathbb{Z}, \quad x \mapsto \#M(x) - \#\overline{M}(x).$$

The class of these functions is

$$\mathrm{GapP} = \{\mathrm{gap}_M \mid M \text{ is a CM.}\}$$

GapP can also equivalently defined as the closure of #P under subtraction (i.e., GapP = #P−#P, where the minus sign denotes elementwise subtraction).

Definition 2.17 (PP [27])**.** A language L is in PP (probabilistic polynomial time) if there exists a function $g \in \mathrm{GapP}$ such that $L = \{x \mid g(x) \geq 0\}$.

Definition 2.18 ($\mathrm{A_0PP}$ [61])**.** A promise problem Π is in $\mathrm{A_0PP}$ (one-sided analogue of AWPP) if there exist functions $t \in \mathrm{FP}$ and $g \in \mathrm{GapP}$ (closure of #P under subtraction) such that

- $\forall x \in \Pi_{\mathrm{yes}} : g(x) > t(x)$,
- $\forall x \in \Pi_{\mathrm{no}} : 0 \leq g(x) \leq \frac{1}{2}t(x)$.

$\mathrm{A_0PP}$ was introduced by Vyalyi [61] to show that QMA = PP implies PH ⊆ PP. The completeness/soundness parameters can be amplified and we can assume that the threshold function t only depends on the length of x.

Lemma 2.19 ([61])**.** *Fix a polynomial $q(n)$. Let $\Pi \in \mathrm{A_0PP}$. There exists $g \in \mathrm{GapP}$ and a polynomial $p(n)$, such that*

- $\forall x \in \Pi_{\mathrm{yes}} : g(x) > 2^{p(|x|)}$,
- $\forall x \in \Pi_{\mathrm{no}} : 0 \leq g(x) \leq 2^{-q(|x|)}2^{p(|x|)}$.

Kuperberg [47] showed that $\mathrm{SBQP} = \mathrm{A_0PP}$.

Definition 2.20 (SBQP [47])**.** A promise problem Π is in SBQP (Small Bounded-Error Quantum Polynomial-Time) if there exists a polynomial-time uniform quantum circuit family $\{Q_n\}$ and a polynomial p such that

- $\forall x \in \Pi_{\mathrm{yes}} : \Pr[Q_{|x|} \text{ accepts } |x\rangle] \geq 2^{-p(n)}$,
- $\forall x \in \Pi_{\mathrm{no}} : \Pr[Q_{|x|} \text{ accepts } |x\rangle] \leq 2^{-p(n)-1}$.

We observe that both $\mathrm{A_0PP}$ and SBQP are in some sense "one-sided" as the range of possible function values or acceptance probabilities in the yes-case is quite large while it is rather close to 0 in the no-case. The promise gap can be exponentially small.

$\mathrm{A_0PP}$ contains the complexity class $\mathrm{C_{=}P}$ [64], which in turn contains SPP [20].

Definition 2.21 ($\mathrm{C_{=}P}$ [27])**.** A language L is in $\mathrm{C_{=}P}$ (Exact-Counting Polynomial-Time) if there exists function $g \in \mathrm{GapP}$ such that $L = \{x \mid g(x) = 0\}$.

Definition 2.22 (SPP [20])**.** A promise problem Π is in SPP (Stoic PP) if there exists a function $g \in \mathrm{GapP}$ such that

- $\forall x \in \Pi_{\mathrm{yes}} : g(x) = 1$,
- $\forall x \in \Pi_{\mathrm{no}} : g(x) = 0$.

2.2.3 Polynomial-Time Hierarchy

Definition 2.23 (PH (Polynomial-Time Hierarchy) [57])**.** Define

$$\Delta_0^{\mathrm{P}} = \Sigma_0^{\mathrm{P}} = \Pi_0^{\mathrm{P}} = \mathrm{P},$$

and for $i \geq 0$

$$\Sigma_{i+1}^{\mathrm{P}} = \mathrm{NP}^{\Sigma_i^{\mathrm{P}}}$$
$$\Pi_{i+1}^{\mathrm{P}} = \mathrm{coNP}^{\Sigma_i^{\mathrm{P}}}$$
$$\Delta_{i+1}^{\mathrm{P}} = \mathrm{P}^{\Sigma_i^{\mathrm{P}}}.$$

Let $\mathrm{PH} = \bigcup_i \Sigma_i^{\mathrm{P}}$.

For example $\Sigma_1^{\mathrm{P}} = \mathrm{NP}$, $\Pi_1^{\mathrm{P}} = \mathrm{coNP}$, and $\Delta_2^{\mathrm{P}} = \mathrm{P}^{\mathrm{NP}}$. Toda's famous theorem states that a single query to a #P oracle is sufficient to decide PH.

Theorem 2.24 (Toda [59])**.** $\mathrm{PH} \subseteq \mathrm{P}^{\#\mathrm{P}[1]}$.

2.2.4 Probabilistically Checkable Proofs

Definition 2.25 ($\mathrm{PCP}[r(n), q(n)]$ [6])**.** A language L is in $\mathrm{PCP}[r(n), q(n)]$ if there exists verifier Turing machine M that behaves as follows:

1. M receives input x, a proof $y \in \{0,1\}^*$ and a random string $z \in \{0,1\}^{r(n)}$ on separate tapes.
2. M computes indices $i_1, \ldots, i_{q(n)}$ without accessing y (it may access z) in polynomial time.
3. M copies proof bits $y_{i_1}, \ldots, y_{i_{q(n)}}$ to its work tape.
4. M accepts or rejects in polynomial time without accessing y.

M must also satisfy the following conditions for all $x \in \{0,1\}^n$:

- If $x \in L$, $\exists y : \Pr_z[M(x,y,z) = 1] = 1$, where $z \in \{0,1\}^{r(n)}$ is chosen uniformly at random.
- If $x \notin L$, $\forall y : \Pr_z[M(x,y,z) = 1] \leq 1/2$.

Theorem 2.26 (PCP Theorem [5])**.** $\mathrm{NP} = \mathrm{PCP}[O(\log(n)), O(1)]$.

Theorem 2.27 ([7])**.** $\mathrm{NEXP} = \mathrm{PCP}[O(\mathrm{poly}(n)), O(\mathrm{poly}(n))]$.

2.3 The Local Hamiltonian Problem

In complexity theory, we are often interested in complete problems for complexity classes. For example, 3-SAT is NP-complete. The local Hamiltonian problem, which we discuss in this section, can be considered its quantum analogue. It is in fact quite straightforward to map any 3-SAT instance to a local Hamiltonian.

Definition 2.28 (k-local Hamiltonian)**.** A Hermitian operator $H \in \mathrm{Herm}\left(\mathcal{B}^{\otimes n}\right)$ acting on n qubits is a k-local Hamiltonian if it can be written as

$$H = \sum_{\substack{S \subseteq [n] \\ |S| \leq k}} H_S \otimes I_{[n] \setminus S}.$$

Additionally, $0 \preccurlyeq H_S \preccurlyeq I$ must hold for all S.

We refer to the minimum eigenvalue $\lambda_{\min}(H)$ as the *ground state energy* of H and the corresponding eigenvectors as *ground states*.

Definition 2.29 (k-LH)**.** Fix a polynomial $p(n)$. The promise problem k-LH (k-local Hamiltonian problem) is defined as follows.

Input:

- A k-local Hamiltonian $H \in \mathrm{Herm}\,(\mathcal{B}^{\otimes n})$.
- Efficiently computable threshold functions $\alpha, \beta : \mathbb{N} \to \mathbb{R}$ satisfying the *promise gap* $\beta(n) - \alpha(n) \geq 1/p(n)$ for all $n \geq 1$.

Output:

YES: $\lambda_{\min}(H) \leq \alpha(n)$
NO: $\lambda_{\min}(H) \geq \beta(n)$

As a warm up, we sketch how to transform a 3-SAT instance into a k-LH instance. Consider the clause $c = (\bar{x}_1 \vee x_2)$. The corresponding 2-local Hamiltonian is simply

$$H_c = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Clearly, $\lambda_{\min}(H_c) = 0$ with ground states $|00\rangle, |01\rangle, |11\rangle$, corresponding to the satisfying assignments of the clause.

In general, H_c is the diagonal matrix that has a 0 at the entries corresponding to satisfying assignments of c and a 1 for unsatisfying assignments. Therefore, all satisfying assignments will be in the nullspace. The Hamiltonian for a conjunction $\Phi = \bigwedge_j c_j$ is then $H_\Phi = \sum_j H_{c_j}$. It can be shown that $\lambda_{\min}(H_\Phi)$ is equal to the minimum number of unsatisfied clauses.

In the following, we prove that k-LH is indeed QMA-complete. We state the proof entirely since we reuse it in parts for our own proofs. The proof taken from Kitaev [43] with some modifications from [22], and adapted slightly.

Theorem 2.30. k-LH *is* QMA*-complete for* $k \geq 5$.

Remark. Kempe and Regev improved this to $k \geq 3$ [40]. Kempe, Kitaev, and Regev subsequently improved this to $k \geq 2$ [39]. For $k = 1$, the problem is in P.

2.3.1 k-LH $\in$ QMA

We begin by showing that k-LH is indeed contained in QMA.

Lemma 2.31. k-LH $\in$ QMA *for* $k = O(\log n)$.

Proof. Let $H \in \mathrm{Herm}\,(\mathcal{B}^{\otimes n})$ be a k-LH instance with thresholds α, β. We construct a QMA-verifier from H.

The general idea is to use the proof $|\psi\rangle$ given to the verifier as potential ground state of H and check if its energy exceeds α. Thus, we need to check if

$$\langle\psi|H|\psi\rangle \leq \alpha(n).$$

To measure $\langle\psi|H|\psi\rangle$, we make use of the fact that H is k-local and can therefore be written as a $H = \sum_{j=1}^{m} H_j$. Hence,

$$\langle\psi|H|\psi\rangle = \sum_{j=1}^{m} \langle\psi|H_j|\psi\rangle.$$

This allows us to compute a value with expectation $\langle\psi|H|\psi\rangle$ by performing the following random experiment:

1. Select $R \in [m]$ uniformly at random.
2. Measure $|\psi\rangle$ using the measurement described by the POVM $\{I - H_R, H_R\}$ corresponding to results $\{0, 1\}$ and output the result.

The given POVM is valid since by definition $0 \preccurlyeq H_R \preccurlyeq I$. It can be implemented with a polynomially sized circuit because H_R only acts on $k = O(\log n)$ qubits [43].

Let X be the random variable corresponding to the output of the experiment. After choosing R, we have in step 2 that

$$\Pr[X = 1 \mid R = r] = \langle\psi|H_r|\psi\rangle.$$

It therefore holds that

$$\begin{aligned}\mathbb{E}[X] &= \Pr[X = 1] \\ &= \sum_{r=1}^{m} \Pr[R = r] \Pr[X = 1 \mid R = r] \\ &= \sum_{r=1}^{m} \frac{1}{m} \langle\psi|H_r|\psi\rangle \\ &= \frac{1}{m} \langle\psi|H|\psi\rangle.\end{aligned}$$

We define a verifier V that runs the above random experiment and accepts if $X = 0$. Hence,

$$\Pr[V \text{ accepts } |\psi\rangle] = 1 - \frac{1}{m} \langle\psi|H|\psi\rangle.$$

It follows that V is a valid QMA verifier with a promise gap of at least $(mp(n))^{-1}$, where $\beta(n) - \alpha(n) \geq p(n)$. □

2.3.2 k-LH is QMA-hard

This direction of the proof is significantly more involved than the previous one. The basic idea is similar to the proof of the well known Cook-Levin Theorem [16, 48]. We construct a Hamiltonian in such a way, that the ground state is a superposition of all intermediate states in the computation.

Consider a promise problem $\Pi \in$ QMA and some instance x. Let $V = V_\ell \cdots V_1$ be the verifier for instances of length $|x|$. V acts on the space

$$\mathcal{B}_A^{\otimes n} \otimes \mathcal{B}_B^{\otimes p(n)} \otimes \mathcal{B}_C^{\otimes q(n)},$$

where p and q are polynomials and registers A, B, and C correspond to input string x, proof $|\psi\rangle$ and ancillae (initialized to $|0\rangle$), respectively. Define

$$P(\psi, x) := (\langle x|_A \langle\psi|_B \langle 0|_C)\, U^\dagger \Pi_{\text{acc}} U\, (|x\rangle_A |\psi\rangle_B |0\rangle_C),$$

where Π_{acc} denotes the projector onto the subspace where the qubit C_1 (i.e., the first qubit of register C) is set to $|1\rangle$. By the definition of QMA, the following holds for an arbitrary inverse polynomial ε, which we choose later:

- If $x \in \Pi_{\text{yes}}$ (i.e., x is a yes-instance), $\exists |\psi\rangle \in \mathcal{B}^{\otimes m} : P(\psi, x) \geq 1 - \varepsilon$.

- If $x \in \Pi_{\text{no}}$, (i.e., x is a no-instance), $\forall |\psi\rangle \in \mathcal{B}^{\otimes m} : P(\psi, x) \leq \varepsilon$.

2.3.2.1 Hamiltonian Construction

We can imagine applying V to some quantum state as applying the gates $V_1, \ldots, V_\ell$ one by one. This gives us *intermediate states*

$$|\psi_t\rangle := V_t \cdots V_1 |x\rangle_A |\psi\rangle_B |0\rangle_C.$$

Our goal now is to construct a Hamiltonian H whose ground state is a superposition of the intermediate states. H acts on the space

$$\mathcal{L} := \mathcal{B}_{ABC}^{\otimes(n+p(n)+q(n))} \otimes \mathbb{C}_D^{\ell+1},$$

where $\mathbb{C}^{\ell+1}$ corresponds to a *clock register* D. The desired low energy state is then the *history state*

$$|\eta\rangle = \frac{1}{\sqrt{\ell+1}} \sum_{t=0}^{\ell} V_t \cdots V_1 |x\rangle_A |\psi\rangle_B |0\rangle_C \otimes |t\rangle_D \in \mathcal{L}, \tag{2.2}$$

for some $|\psi\rangle$ that maximizes $P(\psi, x)$.

Now that we have laid out what the construction of H seeks to achieve, we give its actual definition. H consists of three summands

$$H = H_{\text{in}} + H_{\text{prop}} + H_{\text{out}},$$

"penalizing" vectors diverging from the history state structure.

H_{in} models the condition that the state at timestep 0 has the correct format $|x\rangle_A |\psi\rangle_B |0\rangle_C |0\rangle_D$ for some $|\psi\rangle$.[1] Let $\Pi_j^{(b)}$ be the projector onto the subspace where the j-th qubit equals b.

$$H_{\text{in}} = \left(\sum_{j=1}^{n} \left(\Pi_j^{(\overline{x_j})} \right)_A \otimes I_{BC} + \sum_{j=1}^{q(n)} I_{AB} \otimes \left(\Pi_j^{(0)} \right)_C \right) \otimes |0\rangle\langle 0|_D.$$

Note that H_{in} is not 5-local but rather $O(\log \ell)$-local. We will address that in Section 2.3.2.4.

H_{out} penalizes states $|\eta\rangle$, where the last timestep has a large overlap with $|0\rangle$ on the first qubit of C. Let $\Pi_{\text{rej}} = I - \Pi_{\text{acc}}$,

$$H_{\text{out}} = \Pi_{\text{rej}} \otimes |\ell\rangle\langle\ell|_D.$$

If $|\eta\rangle$ has the structure from (2.2), it holds that

$$\langle\eta| H_{\text{out}} |\eta\rangle = \frac{1 - P(\psi, x)}{\ell + 1}. \tag{2.3}$$

H_{prop} enforces a correct propagation of the initial state to the final state. Specifically, timestep

[1] Kitaev's construction does not use an input register. Instead, x is assumed to be "hardcoded" into the circuit. This construction is taken from [22]

t should result from timestep $t-1$ by applying V_t.

$$H_{\text{prop}} = \sum_{t=1}^{\ell} H_t$$
$$H_t = -V_t \otimes |t\rangle\langle t-1|_D - V_t^\dagger \otimes |t-1\rangle\langle t|_D + I \otimes (|t\rangle\langle t| + |t-1\rangle\langle t-1|)_D$$

2.3.2.2 Completeness

To prove completeness, we must show that $\lambda_{\min}(H)$ is below a certain threshold α if $x \in \Pi_{\text{yes}}$.

Lemma 2.32. *If $x \in \Pi_{\text{yes}}$, then*

$$\lambda_{\min}(H) \leq \frac{\varepsilon}{\ell+1} =: \alpha.$$

Proof. Let $|\eta\rangle$ be as in (2.2). It is easy to show that $|\eta\rangle$ is in the nullspace of H_{in} and H_{prop}. Since $x \in \Pi_{\text{yes}}$, we can choose $|\psi\rangle$ such that $P(\psi, x) \geq 1-\varepsilon$. The lemma then follows from (2.3). □

2.3.2.3 Soundness

To prove soundness, we must show that $\lambda_{\min}(H) \geq \beta$ for $x \in \Pi_{\text{no}}$.

Lemma 2.33. *If $x \in \Pi_{\text{no}}$, then*

$$\lambda_{\min}(H) \geq c(1-\sqrt{\varepsilon})\ell^{-3} =: \beta$$

for some constant $c > 0$.

Note that $\beta - \alpha \geq 1/\text{poly}$ holds if we choose $\varepsilon = O(\ell^{-2})$. It will be helpful to consider H in a different basis to prove the lemma. Specifically, we apply

$$W = \sum_{t=0}^{\ell} V_1^\dagger \cdots V_t^\dagger \otimes |t\rangle\langle t|_D$$

as change of basis operator. The change of basis transforms $|\eta\rangle$ into $|\tilde{\eta}\rangle := W|\eta\rangle$ and H into its conjugate $\tilde{H} = WHW^\dagger$.

We now analyze how the conjugation acts on each term of H. It has no effect on H_{in}:

$$\tilde{H}_{\text{in}} = WH_{\text{in}}W^\dagger = H_{\text{in}}$$

On H_{out}, it conjugates Π_{rej} by $V^\dagger$:

$$\tilde{H}_{\text{out}} = WH_{\text{out}}W^\dagger = \left(V^\dagger \Pi_{\text{rej}} V\right)_{ABC} \otimes |\ell\rangle\langle\ell|_D$$

For H_{prop}, we analyze each H_j term separately. Recall that H_t consists of three summands, the

first of which has the following conjugation:

$$\begin{aligned}
&W\left(V_t \otimes |t\rangle\langle t-1|\right) W^\dagger \\
&\quad = \sum_{r=0}^{\ell}\sum_{s=0}^{\ell}\left(V_1^\dagger \cdots V_r^\dagger \otimes |r\rangle\langle r|\right)\left(V_t \otimes |t\rangle\langle t-1|\right)\left(V_s \cdots V_1 \otimes |s\rangle\langle s|\right) \\
&\quad = \left(V_1^\dagger \cdots V_t^\dagger \cdot V_t \cdot V_{t-1} \cdots V_1\right) \otimes \left(|t\rangle\langle t| \cdot |t\rangle\langle t-1| \cdot |t-1\rangle\langle t-1|\right) \\
&\quad = I \otimes |t\rangle\langle t-1|
\end{aligned}$$

The other two terms are conjugated analogously and we have

$$\begin{aligned}
\tilde{H}_t &= W H_t W^\dagger \\
&= I_{ABC} \otimes \left(|t-1\rangle\langle t-1| - |t-1\rangle\langle t| - |t\rangle\langle t-1| + |t\rangle\langle t|\right)_D \\
&=: I_{ABC} \otimes (E_j)_D,
\end{aligned}$$

and consequently

$$\tilde{H}_{\text{prop}} := W H_{\text{prop}} W^\dagger = I_{ABC} \otimes E_D,$$

where

$$E = \sum_{t=1}^{\ell} E_t = \begin{pmatrix}
1 & -1 & 0 & 0 & 0 & \cdots \\
-1 & 2 & -1 & 0 & 0 & \cdots \\
0 & -1 & 2 & -1 & 0 & \cdots \\
0 & 0 & -1 & 2 & -1 & \cdots \\
0 & 0 & 0 & -1 & \ddots & \ddots \\
\vdots & \vdots & \vdots & \vdots & \ddots & \ddots
\end{pmatrix}$$

E is now an almost diagonal matrix and resembles the transition matrix of a 1D random walk. It can be shown [43] that E has eigenvectors and eigenvalues

$$\begin{aligned}
|\lambda_k\rangle &= \alpha_k \sum_{t=0}^{\ell} \cos\left(q_k\left(t+\frac{1}{2}\right)\right)|t\rangle, \\
\lambda_k &= 2(1-\cos q_k), \\
q_k &= \frac{\pi k}{\ell+1}.
\end{aligned} \tag{2.4}$$

Note that $\lambda_{\min}(H) = \lambda_{\min}(\tilde{H})$ while the latter has a much simpler structure. Unfortunately, not all three summands $\tilde{H}_{\text{in}}, \tilde{H}_{\text{out}}, \tilde{H}_{\text{prop}}$ commute pairwise, which makes the eigenvalue analysis difficult. At least $\tilde{H}_{\text{in}}$ and $\tilde{H}_{\text{out}}$ commute. Therefore, we partition $\tilde{H} = A_1 + A_2$ with $A_1 = \tilde{H}_{\text{in}} + \tilde{H}_{\text{out}}$ and $A_2 = \tilde{H}_{\text{prop}}$. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be their respective nullspaces (i.e., $\mathcal{L}_i = \text{Null}(A_i)$). We will later show that $\mathcal{L}_1 \cap \mathcal{L}_2 = 0$. Therefore, $\tilde{H} \succ 0$. By obtaining bounds for the nonzero eigenvalues of A_1 and A_2, we can use the below lemma to give a lower bound for $\lambda_{\min}(\tilde{H})$. The lemma requires the angle $\angle(\mathcal{L}_1, \mathcal{L}_2)$ between the two nullspaces:

$$\angle(\mathcal{L}_1, \mathcal{L}_2) := \cos^{-1} \max_{\substack{|\eta_1\rangle \in \mathcal{L}_1 \\ |\eta_2\rangle \in \mathcal{L}_2}} |\langle \eta_1 | \eta_2 \rangle|$$

Lemma 2.34 ([43])**.** *Let A_1, A_2 be nonnegative operators with nullspaces $\mathcal{L}_1, \mathcal{L}_2$ such that $\mathcal{L}_1 \cap \mathcal{L}_2 =$*

0. *Let v be a lower bound on the smallest nonzero eigenvalue of A_1 and A_2. Then*

$$\lambda_{\min}(A_1 + A_2) \geq 2v \sin^2 \frac{\angle(\mathcal{L}_1, \mathcal{L}_2)}{2}.$$

To apply the lemma, we need to determine a suitable lower bound v. A_1 has only nonnegative integer eigenvalues since it is the sum of pairwise commuting projectors.

A_2 clearly has the same eigenvalues as E whose smallest positive eigenvalue is λ_1 as defined in (2.4).

$$\lambda_1 = 2\left(1 - \cos\frac{\pi}{\ell+1}\right) \geq c'\ell^{-2} =: v, \tag{2.5}$$

where c' is some positive constant and the inequality follows from the Taylor expansion of $\cos x - 1$.

Next, we provide a lower bound for the angle $\sin^2\theta$ with $\theta = \angle(\mathcal{L}_1, \mathcal{L}_2)$.

Lemma 2.35 ([22, 43])**.**

$$\sin^2\theta \geq \frac{1-\sqrt{\varepsilon}}{\ell+1}$$

Proof. We simplify this calculation by computing an upper bound for

$$\cos^2\theta = \max_{\substack{|\eta_1\rangle\in\mathcal{L}_1\\ |\eta_2\rangle\in\mathcal{L}_2}} |\langle\eta_1|\eta_2\rangle|^2 = \max_{|\eta_2\rangle\in\mathcal{L}_2} \langle\eta_2|\Pi_{\mathcal{L}_1}|\eta_2\rangle, \tag{2.6}$$

where $\Pi_{\mathcal{L}_2}$ denotes the projector onto $\mathcal{L}_2$.

It can easily be shown that

$$\begin{aligned}
\mathcal{L}_1 &= \mathcal{K}_1 \oplus \mathcal{K}_2 \oplus \mathcal{K}_3,\\
\mathcal{K}_1 &= |x\rangle_A \otimes \mathcal{B}_B^{\otimes p(n)} \otimes |0\rangle_C \otimes |0\rangle_D,\\
\mathcal{K}_2 &= \mathcal{B}_{ABC}^{\otimes(n+p(n)+q(n))} \otimes \mathrm{Span}\left(|1\rangle, \ldots, |\ell-1\rangle\right)_D,\\
\mathcal{K}_3 &= V^\dagger\left(|1\rangle_{C_1} \otimes \mathcal{B}^{\otimes(n+p(n)+q(n)-1)}\right)_{ABC} \otimes |\ell\rangle_D,
\end{aligned}$$

where $\oplus$ denotes the direct sum of vector spaces (i.e., the sum of orthogonal vector spaces) and

$$\begin{aligned}
\mathcal{L}_2 &:= \mathcal{B}^{\otimes(n+p(n)+q(n))} \otimes |\gamma\rangle,\\
|\gamma\rangle &:= |\lambda_0\rangle = \frac{1}{\sqrt{\ell+1}}\sum_{t=0}^{\ell}|t\rangle.
\end{aligned}$$

Note that we slightly abuse notation here by using $|x\rangle$ to represent the span of the vector $|x\rangle$. Since $\mathcal{L}_1$ is the direct sum of $\mathcal{K}_1, \mathcal{K}_2, \mathcal{K}_3$, it holds that

$$\Pi_{\mathcal{L}_1} = \Pi_{\mathcal{K}_1} + \Pi_{\mathcal{K}_2} + \Pi_{\mathcal{K}_3}.$$

Let $|\eta_2\rangle \in \mathcal{L}_2$. We analyze the terms $\langle\eta_2|\Pi_{\mathcal{K}_j}|\eta_2\rangle$ separately.

It can easily be shown that

$$\langle\eta_2|\Pi_{\mathcal{K}_1}|\eta_2\rangle \leq \frac{\ell-1}{\ell+1}.$$

Since $|\eta_2\rangle \in \mathcal{L}_2$, we can write $|\eta_2\rangle = |\beta\rangle_{ABC} \otimes |\gamma\rangle_D$. It follows that

$$\begin{aligned}\langle\eta_2| \left(\Pi_{\mathcal{K}_1} + \Pi_{\mathcal{K}_3}\right) |\eta_2\rangle &\leq \frac{1}{\ell+1} \langle\beta| \left(\Pi_{\mathcal{X}} + \Pi_{\mathcal{Y}}\right) |\beta\rangle \\ &\leq \frac{1+\cos\alpha}{\ell+1},\end{aligned}$$

where $\mathcal{X} = |x\rangle_A \otimes \mathcal{B}_B^{\otimes p(n)} \otimes |0\rangle_C$, $\mathcal{Y} = V^\dagger \left(|1\rangle_{C_1} \otimes \mathcal{B}^{\otimes(n+p(n)+q(n)-1)}\right)_{ABC}$ and $\alpha = \angle(\mathcal{X}, \mathcal{Y})$. The second inequality follows from the (omitted) proof of Lemma 2.34.

We can interpret $\mathcal{X}$ as the space of valid initial states and $\mathcal{Y}$ as the space of states that V accepts perfectly. Recalling (2.6), it follows that $\cos^2\alpha \leq \varepsilon$ since $x \in \Pi_{\text{no}}$.

Thus, we have

$$\langle\eta_2|\Pi_{\mathcal{L}_1}|\eta_2\rangle \leq \frac{\ell-1}{\ell+1} + \frac{1+\sqrt{\varepsilon}}{\ell+1} = 1 - \frac{1-\sqrt{\varepsilon}}{\ell+1}.$$

The lemma follows due to $\sin^2\theta = 1 - \cos^2\theta$. □

Lemma 2.33 is now easily proven by applying Lemma 2.34 to the bounds from Lemma 2.35 and (2.5).

2.3.2.4 Locality

Our above proofs for completeness and soundness imply QMA-hardness of k-LH for $k = \Omega(\log n)$ since the clock requires $\Theta(\log n)$ qubits. We need to modify our construction to make it 5-local. To that end, we choose to represent the clock in "unary". This means that the clock value is represented by the number of leading 1's in the bitstring. Concretely, we embed the clock register $\mathbb{C}^{\ell+1}$ in the larger space $\mathcal{B}^{\otimes\ell}$ using the embedding

$$f : \mathbb{C}^{\ell+1} \to \mathcal{B}^{\otimes\ell},\ |t\rangle \mapsto |1^t 0^{\ell-t}\rangle.$$

Assuming the time register follows this embedding, to check timestep t, we only need to check that qubit t is set to $|1\rangle$ and $t+1$ is set to $|0\rangle$.

Therefore, we replace the operators on the D register in the definition of H (see Section 2.3.2.1) with the following 3-local operators:

$$\begin{aligned}|0\rangle\langle 0| &\to \Pi_1^{(0)}, & |0\rangle\langle 1| &\to (|0\rangle\langle 1|)_1 \Pi_2^{(0)} \\ |t\rangle\langle t| &\to \Pi_t^{(1)} \Pi_{t+1}^{(0)}, & |t-1\rangle\langle t| &\to \Pi_{t-1}^{(1)} (|0\rangle\langle 1|)_t \Pi_{t+1}^{(0)} \\ |\ell\rangle\langle\ell| &\to \Pi_\ell^{(1)}, & |\ell-1\rangle\langle\ell| &\to \Pi_{\ell-1}^{(1)} (|0\rangle\langle 1|)_\ell\end{aligned}$$

Note that $(|0\rangle\langle 1|)_t$ denotes the operator that acts as $|0\rangle\langle 1|$ on qubit t and identity on the remaining qubits.

By performing these replacements on H, we obtain a new Hamiltonian $H_{\text{ext}} \in \text{Herm}(\mathcal{L}_{\text{ext}})$, where $\mathcal{L}_{\text{ext}} = \mathcal{B}_{ABC}^{n+p(n)+q(n)} \otimes \mathcal{B}_D^\ell$ is the space obtained from $\mathcal{L}$ by replacing the D register with $\mathcal{B}^\ell$. Using the previously defined embedding, we can say $\mathcal{L} \subseteq \mathcal{L}_{\text{ext}}$. H_{ext} maps $\mathcal{L}$ to itself and acts on $\mathcal{L}$ in the same way as H. That means

$$\forall|\psi\rangle \in \mathcal{L} :\ H_{\text{ext}} f(|\psi\rangle) = f(H|\psi\rangle). \tag{2.7}$$

It remains the problem that H_{ext} only "works correctly" on $\mathcal{L}$. Vectors with incorrectly formatted clock (e.g., $|0^{\ell-1}1\rangle$) are in the nullspace of H_{ext}. Therefore, we introduce a new Hamiltonian term to penalize incorrectly formatted clock registers (i.e., 01 occurs in the bitstring).

$$H_{\text{stab}} = I_{ABC} \otimes \sum_{t=1}^{\ell-1} \Pi_t^{(0)} \Pi_{t+1}^{(1)}$$

We now use $H' := H_{\text{ext}} + H_{\text{stab}}$. Since $\text{Null}(H_{\text{stab}}) = \mathcal{L}$, the completeness proof Lemma 2.32 for H still works for H'.

To show soundness, we need to argue that $\lambda_{\min}(H') \geq c(1-\sqrt{\varepsilon})\ell^{-3}$. Since H' and H_{ext} map $\mathcal{L}$ to itself, we can analyze its smallest eigenvalue in $\mathcal{L}$ and its orthogonal complement $\mathcal{L}^{\perp}$ independently.

On $\mathcal{L}$, we may apply Lemma 2.33 due to (2.7), which implies

$$\lambda_{\min}(H_{\text{ext}}) \geq c(1 - \sqrt{\varepsilon})\ell^{-3}.$$

$\lambda_{\min}(H_{\text{stab}}) = 0$ on $\mathcal{L}$ since $\mathcal{L} = \text{Null}(H_{\text{stab}})$.

On $\mathcal{L}^{\perp}$ it holds that $\lambda_{\min}(H_{\text{ext}}) = 0$ and $\lambda_{\min}(H_{\text{stab}}) \geq 1$. The latter inequality follows from the fact that H_{stab} only has nonnegative integer eigenvalues since it is the sum of commuting projectors.

The proof of Theorem 2.30 is now complete.

Chapter 3

$\mathrm{P}^{\mathrm{QMA}}$ with Tree-like Dependencies

Suppose we have a polynomial-time Turing machine with oracle access to QMA. Each query the Turing machine asks the oracle can depend on previous queries. We can model this as a DAG whose nodes represent queries and edges dependencies. If node v has an incoming edge from u, then v's query depends on u's. How does the complexity change if we restrict the DAG to trees? We call the resulting complexity class TREES(QMA) (formally defined in Definition 3.19). It turns out that TREES(QMA) = $\mathrm{P}^{\mathrm{QMA[log]}}$, which we prove in this chapter.

This result is essentially a quantum analogue of Gottlob's [31] analogous result for NP. We build upon his proof to prove our result. Therefore, we present Gottlob's proof for $\mathrm{P}^{\mathrm{NP[log]}}$ = TREES(NP) in Section 3.1.

We will give two proofs TREES(QMA) = $\mathrm{P}^{\mathrm{QMA[log]}}$. The first reduces TREES(QMA) to APX-SIM by constructing a query Hamiltonian (similar to Ambainis' proof for APX-SIM being $\mathrm{P}^{\mathrm{QMA[log]}}$-complete [4]; see Section 3.3). The second proof uses a *total solution weight* function (similar to (3.5); see Section 3.4). Afterwards, we generalize the result to MA, QCMA, and QMA(2) (see Section 3.5).

Finally, we investigate whether we can go beyond tree-like dependencies. For P^{NP}, we prove that the dependency graph having bounded tree-width leads to containment in $\mathrm{P}^{\mathrm{NP[log]}}$ (see Section 3.6).

3.1 P^{NP} with Tree-Like Dependencies

In this section, we will show that P^{NP} with tree-like query dependencies equals $\mathrm{P}^{\mathrm{NP[log]}}$. We choose to represent the problem differently from Gottlob as to make it easier to adapt to the quantum setting. The proofs remain very similar to those given in [31].

3.1.1 Formal Definition

First, we define the language NP-DAG to model the problem where a set of queries to an NP-oracle are described in a DAG. Each node represents a query that depends on results of the queries from direct predecessors.

Definition 3.1 (NP-DAG)**.** An NP-DAG instance is defined by an n-node DAG $G = (V = \{v_1, \ldots, v_n\}, E)$. $v_n \in V$ is the unique node with $\mathrm{outdeg}(v_n) = 0$, which we denote the *result node*. Each node $v_i \in V$ contains the description of a polynomially-sized circuit C_i with designated input and proof register (akin to an NP-verifier). The input register's size equals $\mathrm{indeg}(v_i)$. C_i

implicitly defines a language L_i (L_i contains the outputs of previous queries for which there exists an accepting proof for C_i). By padding L_i with 0^n, we get $L_i \in \text{NP}$ (v_i might have very few inputs but C_i may require a proof of size $\text{poly}(n)$).

We say $G \in \text{NP-DAG}$ if the verification procedure (see Algorithm 3.1) accepts G, which means $\textsc{Verify}(G) = 1$.

Algorithm 3.1 Verification procedure for NP-DAG

function $\textsc{Verify}(G = (V, E))$
  Sort the nodes of V topologically into $v_1, \ldots, v_n$.
  Denote by $x_i \in \{0,1\}$ the result of v_i's query.
  for $i = 1, \ldots, n$ **do**
    $z_i \leftarrow \bigcirc_{v_j \in \text{pred}(v_i)} x_j$ ▷ The concatenation order is specified by C_i.
    $x_i \leftarrow \begin{cases} 1, & \text{if } z_i \in L_i \\ 0, & \text{otherwise} \end{cases}$
  return x_n

We call a string $x \in \{0,1\}^n$ that can be constructed by this procedure a *correct query string*.

Note that the language NP-DAG corresponds to Gottlob's DAGS(SAT) (refer to [31] for the formal definition). The main difference is that NP-DAG models each query as a circuit that receives the previous results as input (imagine the edges as *sending* the output of one circuit to the input of the next), whereas each node in DAGS(SAT) contains a boolean formula. These formulas are connected with *linking variables* that are set to 1 iff the associated formula is satisfiable. It is easy to see that both definitions are equivalent. The reason for the divergent definition is that we want to replace NP with QMA, and consequently classical circuits and proofs with quantum circuits and proofs.

Lemma 3.2. NP-DAG *is* P^{NP}*-complete.*

Proof. $\text{NP-DAG} \subseteq \text{P}^{\text{NP}}$ holds because the verification procedure defined in in Definition 3.1 can easily be executed by a Turing machine with an NP-oracle. The oracle is used to compute the x_i values. The remaining operations are trivial.

To argue P^{NP}-hardness of NP-DAG, we describe the polynomial-time reduction from P^{NP} to NP-DAG. Let M be some P^{NP} machine and x an input. Without loss of generality, we may assume that M always performs $m \leq \text{poly}(|x|)$ queries. Let G be a DAG with m nodes. Node v_i represents the i-th query of M and has incoming edges from $v_1, \ldots, v_{i-1}$. C_i is then defined as the circuit that first computes the state of M prior to the i-th query using the results of all previous queries and then accepts iff that query is accepted by the oracle. By construction, $G \in \text{NP-DAG}$ iff M accepts x. □

By requiring G to be a tree, we obtain the complexity class TREES(NP).

Definition 3.3 (TREES(NP))**.** The complexity class TREES(NP) is the class of all languages that are polynomial-time reducible to NP-DAG, such that G of the resulting NP-DAG instance is a tree.

3.1.2 $\text{TREES(NP)} = \text{P}^{\text{NP[log]}}$

We will now prove the following theorem.

Theorem 3.4. $\text{TREES(NP)} = \text{P}^{\text{NP[log]}}$.

The first direction is straightforward:

Lemma 3.5. $\text{TREES(NP)} \supseteq \text{P}^{\text{NP[log]}}$.

Proof. Let x be an input to a $\text{P}^{\text{NP[log]}}$ machine M with $n = |x|$. Since M performs only $O(\log n)$ oracle queries, each query can depend on at most $O(\log(n))$ other queries. Therefore, there are only poly(n) possibilities for each query. We construct G by having a node for each possible query without any inputs and an output node that receives the results of all possible queries and then simulates M. It holds that $G \in$ NP-DAG iff M accepts x. □

In the following, we will show the other direction, $\text{TREES(NP)} \subseteq \text{P}^{\text{NP[log]}}$.

3.1.2.1 Weighting Functions

We begin by introducing the concept of *admissable weighting functions*.

Definition 3.6. Let $G = (V, E)$ be a DAG.

- For each $v \in V$, denote by $v\uparrow$ the set of all nodes reachable from v by a directed path of length ≥ 1.
- A *weighting function* for G is a function $\nu : V \to \mathbb{N}$.
- Let $c \geq 1$ be a constant. A weighting function ν is c-admissable iff

$$\forall v \in V : \nu(v) > c \sum_{w \in v\uparrow} \nu(w). \tag{3.1}$$

- The total weight $W_\nu(G)$ of G under weighting function ν is defined as

$$W_\nu(G) = \sum_{v \in V} \nu(v).$$

- A weighting function ν is *polynomial-time computable* iff $\nu(v)$ and $W_\nu(G)$ can be computed in polynomial time for all graphs G and $v \in V$.

Note that Gottlob does not define c-admissable weighting functions. We may omit c in the following for $c = 1$, which corresponds to Gottlob's definition. We define two general families of c-admissable weighting functions.

Definition 3.7. Let $G = (V, E)$ be a DAG. We divide G recursively into levels. Level 0 is made up by the nodes without incoming edges. Level $i + 1$ contains nodes v that have a only inputs w (i.e., $(w, v) \in E$) with level$(w) \leq i$ and at least one input w with level$(w) = i$. We denote the level of a node $v \in V$ by level(v). Nodes on the last level are called *terminal nodes*. The *depth* of G (depth(G)) corresponds to the maximum level number.

Lemma 3.8. *For any DAG $G = (V, E)$, the weighting functions ρ and ω defined below are c-admissable and polynomial-time computable.*

$$\rho(v) = (c|V|)^{\text{depth}(G) - \text{level}(v)}$$
$$\omega(v) = (2c)^{|v\uparrow|}$$

Proof. We begin by proving c-admissibility for ρ. Let $v \in V$. Then

$$\begin{aligned} c \sum_{w \in v\uparrow} \nu(w) &< c|V|(c|V|)^{\mathrm{depth}(G)-\mathrm{level}(v)+1} \\ &= (c|V|)^{\mathrm{depth}(G)-\mathrm{level}(v)} = \rho(v). \end{aligned}$$

The inequality follows from the fact that there are less than $|V|$ summands and all $w \in v\uparrow$ are at a higher level than v.

To prove c-admissibility for ω, we introduce the notion of *strata*. Stratum 0 consists of all terminal nodes (i.e., without outgoing edges). Stratum i consists of those nodes whose outgoing edges all point to strata $< i$ and at least one to a node in stratum $i-1$. We show that ω satisfies the condition (3.1) by induction on the strata. For Stratum 0, the condition holds trivially since it only consists of terminal nodes.

Now assume (3.1) holds for all strata $\leq i$. We show that it will also hold for stratum $i+1$. Note that there may exist a node $w \in \mathrm{suc}(v)$ that is also reachable from another successor $w' \in \mathrm{suc}(v)$ (i.e., $w \in w'\uparrow$). To exclude those successors, we define *pure successors*

$$\mathrm{puresuc}(v) = \mathrm{suc}(v) \setminus \{w \in \mathrm{suc}(w) \mid \exists w' \in \mathrm{suc}(v) : w \in w'\uparrow\}.$$

It trivially holds for all $v \in V$ that

$$v\uparrow = \mathrm{puresuc}(v) \uplus \bigcup_{w \in \mathrm{puresuc}(v)} w\uparrow. \tag{3.2}$$

Since G is acyclic, v cannot have two successors that are both reachable from another and therefore $\mathrm{puresuc}(v) \neq \emptyset$. We write $\mathrm{puresuc}(v) = \{w_1, \ldots, w_m\}$ for some $m \geq 1$. Let $u = \max_{i \in [m]} |w_i\uparrow|$. Due to (3.2), we have $|v\uparrow| \geq m + u$ and thus

$$\omega(v) = (2c)^{|v\uparrow|} \geq (2c)^{m+u} = (2c)^m \cdot (2c)^u \geq 2cm \cdot (2c)^u. \tag{3.3}$$

By the induction hypothesis, for all $i \in [m]$

$$(2c)^u \geq \omega(w_i) > c \sum_{x \in w_i\uparrow} \omega(x). \tag{3.4}$$

Therefore, with (3.2) and (3.3), it follows that

$$\begin{aligned} \sum_{w \in v\uparrow} \omega(w) &\leq \sum_{i=1}^{m} \left(\omega(w_i) + \sum_{x \in w_i\uparrow} \omega(x) \right) && \text{(apply (3.2))} \\ &\leq \sum_{i=1}^{m} 2\omega(w_i) && \text{(apply (3.4))} \\ &< 2m \cdot (2c)^u && \text{(apply (3.4))} \\ &\leq \frac{1}{c}\omega(v), && \text{(apply (3.3))} \end{aligned}$$

which completes the proof for stratum $i+1$.

Both functions are trivially polynomial-time computable since their values are bounded by $2^{\mathrm{poly}(|V|)}$. □

3.1.2.2 Solving NP-DAG for Bounded Total Weight

In the following, we will show how to use binary search to solve an NP-DAG instance G with $O(\log W_\nu(G))$ NP-oracle queries, where ν is an admissable polynomial-time weighting function.

For the binary search, we define the *total solution weight* function. Let $G = (V = \{v_1, \ldots, v_n\}, E)$ be an NP-DAG instance. We assume without loss of generality that each circuit C_i receives a proof $y_i \in \{0,1\}^m$. The total solution weight is then given by

$$\begin{aligned} &t : \{0,1\}^n \times (\{0,1\}^m)^n \to \mathbb{Z}, \\ &\quad t(x, y_1, \ldots, y_n) = \begin{cases} \sum_{i=1}^n \nu(v_i)\, C_i(z_i, y_i), & \text{if } \forall i \in [n] : x_i = C_i(z_i, y_i) \\ -1, & \text{otherwise} \end{cases}, \end{aligned} \tag{3.5}$$

where the z_i are defined as in Definition 3.1. The condition checks that all x_i correspond to the outputs of C_i. Let $T := \max_{x,y} t(x, y)$.

Lemma 3.9. *The question whether $T > s$ holds is in* NP *for any integer s.*

Proof. The NP-verifier receives x, y as proof and accepts if $t(x, y) > s$, which can easily be computed in polynomial time. Correctness is then obvious. □

The lemma allows us to compute T in $O(\log W_\nu(G))$ NP-oracle queries. Next, we argue that $t(x, y) = T$ implies that x is a correct query string. This allows us to determine x_n with an additional NP-query (recalling that x_n corresponds to the output of the result node).

Lemma 3.10. *If $t(x, y) = T$, then x is a correct query string.*

Proof. Assume there exist x, y with $t(x, y) = T \geq 0$. Let x_i be an incorrect entry in the query string. $x_i = 1$ is only possible if $z_i \in L$. Since $x_i = C_i(z_i, y_i)$, it is not possible to have an incorrect $x_i = 1$. Hence, $x_i = 0$, but $z_i \in L_i$. Therefore, there exists a $y_i' \neq y_i$ such that $C_i(z_i, y_i') = 1$.

Define y' by replacing the i-th entry in y with y_i' and $x_j' = C_j(z_j', y_j')$ for $j \in [n]$. The z_i' are computed from x' in the same way as z_i is computed from x. Then,

$$\begin{aligned} t(x', y') - t(x, y) &= \sum_{j=1}^n \nu(v_j)\, C_j(z_j', y_j') - \sum_{j=1}^n \nu(v_j)\, C_j(z_j, y_j) \\ &= \nu(v_i)\big(C_i(z_i', y_i') - C_i(z_i, y_i)\big) + \sum_{v_j \in v_i\uparrow} \nu(v_j)\, \big(C_i(z_j', y_j') - C_j(z_j, y_j)\big) \\ &\geq \nu(v_i) - \sum_{v_j \in v_i\uparrow} \nu(v_j) \\ &\geq 1. \end{aligned}$$

The second equality follows from the fact that we only modified x_i and y_i for v_i and its successors $v_i\uparrow$. The last inequality follows from (3.1) since ν is admissable. □

The next theorem follows immediately.

Theorem 3.11. *Let G be an* NP-DAG *instance and ν an admissable polynomial-time weighting function. A Turing machine can decide if $G \in$ NP-DAG using $\lceil \log W_\nu(G) \rceil + 1$ NP-oracle queries.*

Corollary 3.12. *If $\operatorname{depth}(G) = O(\log^i n)$, then $O(\log^{i+1} n)$ queries suffice.*

Proof. We use ρ as weighting function. Then,

$$W_\rho(G) \leq n \cdot n^{\mathrm{depth}(G)} = n^{O(\log^i n)} = 2^{O(\log^{i+1} n)}.$$

Applying Theorem 3.11 concludes the proof. □

3.1.2.3 Flattening the Tree

Unfortunately, $W_\nu(G)$ may be exponential for any admissable weighting function, even if T is a tree (e.g., if G is a simple path). Therefore, our goal is to "flatten" the tree such that each node in the resulting DAG G has $O(\log n)$ successors. Then $W_\omega(G) \leq \mathrm{poly}(n)$.

The essential operation for flattening the tree will be the notion of a *hypertree*. It is defined similarly to the NP-DAG with the main difference that each node contains multiple queries and the output of each node is determined by an additional output circuit.

Definition 3.13 (NP-HYPERTREE)**.** An NP-HYPERTREE instance is defined by an n-node rooted tree $T = (V = \{v_1, \ldots, v_n\}, E)$. v_n is the root with $\mathrm{outdeg}(v_n) = 0$. Each node v_i specifies:

- Descriptions of polynomial-time circuits $\mathrm{circuits}(v_i) = \{C_{i,1}, \ldots C_{i,m_i}\}$. Each circuit $C_{i,j}$ has the following properties:
  - $C_{i,j}$ specifies a subset of hypernodes $\mathrm{inputs}(C_{i,j}) \subseteq \mathrm{pred}(v_i)$ as inputs.
  - $C_{i,j}$ has designated input and proof registers (akin to an NP-verifier). The input register has size $|\mathrm{inputs}(C_{i,j})|$.
  - $C_{i,j}$ implicitly defines a language $L_{i,j} \in \mathrm{NP}$.
- Description of a polynomial-time output circuit $\mathrm{output}(v_i) = O_i$ with a single input register of size $|m_i|$ (i.e., *not* an NP-verifier).

The hypertree T can also be viewed as an NP-DAG instance by treating the circuits $C_{i,j}$ and O_i as individual nodes with

$$\mathrm{pred}(C_{i,j}) = \{O_k \mid v_k \in \mathrm{inputs}(C_{i,j})\}$$

and $\mathrm{pred}(O_i) = \mathrm{circuits}(v_i)$. We denote this graph by $\mathrm{DAG}(T)$. We say $T \in$ NP-HYPERTREE iff $\mathrm{DAG}(T) \in$ NP-DAG.

Remark. In $\mathrm{DAG}(T)$, the O_i circuits are not queries and do not require a proof. Therefore, $O_1, \ldots, O_{n-1}$ can be "inlined" into their direct successors.

Let G be an NP-DAG instance that is a tree. We can easily transform G into a hypertree T, such that $G \in$ NP-DAG iff $T \in$ NP-HYPERTREE. We do this by transforming each node v_i of G into a hypernode that contains the single circuit C_i and $O_i := \mathrm{id}$.

Next, we introduce the *fusion* operation for hypertrees: The underlying idea is that we can flatten the tree by computing the result of a circuit before we know all of its inputs. We select some input of a circuit and replace the circuit with two copies, in which that input is replaced with a constant 0 or 1. Once the input is known, we select which output to use.

Definition 3.14 (Fusion)**.** Let $T = (V, E)$ be a hypertree with nodes v and w such that $(w, v) \in E$ and v is a *singleton* hypernode (i.e., it contains a single circuit C_v). The *fusion* $w \triangleleft v$ is a new hypernode with the following properties:

- $\mathrm{circuits}(w \triangleleft v) = \mathrm{circuits}(w) \cup \{C_v^0, C_v^1\}$, where C_v^b is the circuit obtained from C_v by replacing the input from w with a constant b.

- $\text{pred}(w \triangleleft v) = \text{pred}(v) \cup \text{pred}(w)$.
- $\text{output}(w \triangleleft v) = O_{w \triangleleft v}$ where $O_{w \triangleleft v}$ is the circuit that computes b as the output of C_w and then returns the output of C_v^b.

Let $T(w \triangleleft v)$ denote the hypertree obtained by replacing v and w in T by $w \triangleleft v$.

An example of a fusion is depicted in Figure 3.1.

Lemma 3.15. *T and $T(w \triangleleft v)$ are equivalent, i.e.,*

$$T \in \text{NP-HYPERTREE} \quad \Leftrightarrow \quad T(w \triangleleft v) \in \text{NP-HYPERTREE}.$$

Proof. Follows directly from Definition 3.14. □

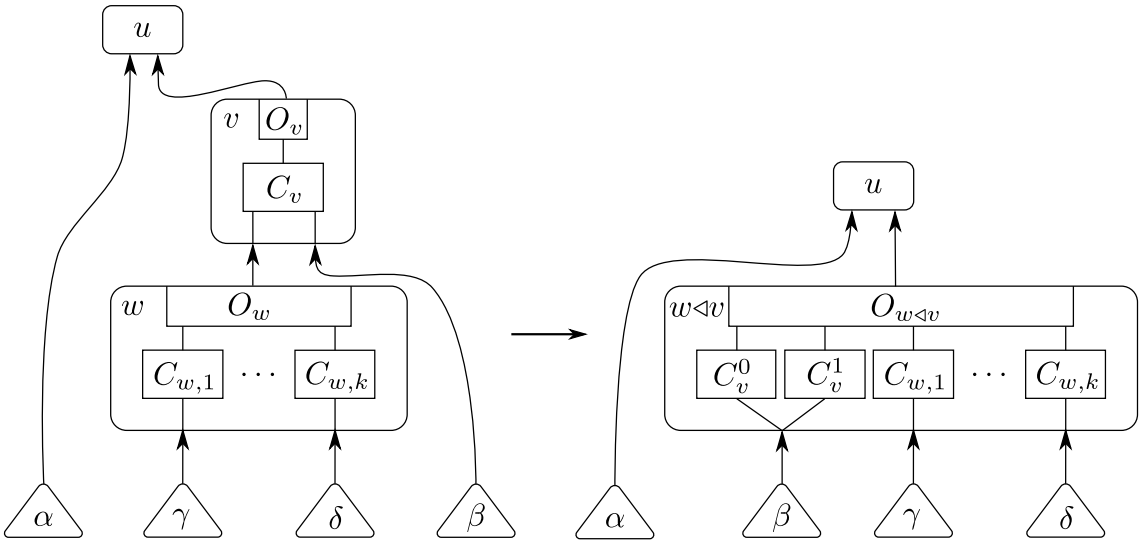


Figure 3.1: Example of the fusion $w \triangleleft v$ (the triangles represent subtrees).

Using the fusion operation, we will "flatten" the hypertree. For that, we introduce the notion of *bad hypernodes*. Let $T = (V, E)$ be a hypertree, and $v \in V$ with $\text{level}(v) = \ell$. We say v is *bad* if it has exactly one input (or direct predecessor) w on level $\ell - 1$. Note that v may have multiple inputs on levels $\leq \ell - 2$ and it cannot be a bottom hypernode (i.e., on level 0). The idea is, that if a hypertree has no bad hypernodes, then every node has at least 2 inputs in the previous level and $\text{depth}(T)$ must be logarithmic. For this, we use the Squeeze(T) function from Algorithm 3.2.

We now argue that Squeeze behaves correctly.

Lemma 3.16. *Let G be an n-node* NP-DAG *instance that is a tree, and $T =$ Squeeze(G).*

(1) Each v selected in Squeeze is a singleton hypernode.

(2) Squeeze(G) terminates and T is equivalent to G.

(3) T has at most n hypernodes.

(4) $\text{depth}(T) \leq \lceil \log n \rceil$.

Proof. (1) We argue inductively that during the execution of Squeeze(G), composite hypernodes (i.e., with multiple circuits) are never bad. Initially, this holds because there are no composite hypernodes. Now let v be the selected bad hypernode. Since BadHypernode returns a bad hypernode that minimizes $\text{level}(v)$, we know that w cannot be bad. Therefore, w is either a bottom

Algorithm 3.2 Given NP-DAG instance G, where G is a tree, compute an equivalent hypertree without bad hypernodes.

```
1: function SQUEEZE(G)
2:     Convert G into the equivalent hypertree T = (V, E).
3:     while T has a bad hypernode do
4:         v ← BADHYPERNODE(T)
5:         Let w ∈ pred(v) with level(w) = level(v) − 1.        ▷ w is unique
6:         T ← T(w◁v)                                           ▷ See Figure 3.1
7:     return T

8: function BADHYPERNODE(T = (V, E))
9:     return bad hypernode v ∈ V that minimizes level(v).
```

hypernode, in which case $w \triangleleft v$ will also be a bottom hypernode, or it has at least two inputs at level $\text{level}(w) - 1$. These will become inputs of $w \triangleleft v$ at level $\text{level}(w \triangleleft v) - 1$. Hence, $w \triangleleft v$ is not a bad hypernode. The subtree of all predecessors of $w \triangleleft v$ does not contain any bad hypernodes and therefore will not be modified in future iterations. Thus, $w \triangleleft v$ will not become bad.

(2) Each iteration reduces the number number of hypernodes by 1. Therefore, SQUEEZE(G) terminates and T has at most n hypernodes (3). T is equivalent to G due to Lemma 3.15.

(4) T has no bad hypernodes by construction. Therefore, every hypernode v has at least two direct predecessors on level $\text{level}(v) - 1$. Since T is a tree, it hast at least $2^{\text{depth}(T)+1} - 1$ nodes. □

We finally argue that each node in DAG(T) has $O(\log n)$ successors which allows us to apply Theorem 3.11 with $\nu = \omega$.

Lemma 3.17. *Let G be an n-node* NP-DAG *instance that is a tree, $T =$ SQUEEZE(G), and $G' = (V', E') = \text{DAG}(T)$. It holds for all $v \in V'$ that $|v{\uparrow}| = O(\log n)$.*

Proof. We say that a circuit C' in a hypertree T' *depends* on another circuit C, if there exists a directed path from C to C' in DAG(T'). We will argue inductively that during the execution of SQUEEZE(G), each circuit C in T has at most 2 non-output circuits C' in each level that depend on C.

Initially, this clearly holds since DAG(T) is still a tree. Now consider a fusion $w \triangleleft v$ as depicted in Figure 3.1. Note that the dependent circuits are only modified for subtree β. After the fusion, only circuits C_v^0 and C_v^1 depend on β in $\text{level}(w \triangleleft v)$. Afterwards, the entire subtree of $w \triangleleft v$ will not be modified further as argued in the proof of Lemma 3.16.

Trivially, each circuit C has at most 1 output circuit in each layer that depends on C. The lemma then follows due to $\text{depth}(T) \leq \lceil \log n \rceil$. □

Lemma 3.17 implies the following theorem.

Theorem 3.18. TREES(NP) = DAGS(NP).

3.2 Formal Definition

Similarly to Section 3.1.1, we begin by defining the complexity class QMA-DAG as a quantum analogue to NP-DAG. We cannot define QMA-DAG in the exact same way as NP-DAG because we are dealing with promise problems. Therefore, the verification procedure becomes nondeterministic.

Definition 3.19 (QMA-DAG)**.** A QMA-DAG instance is defined by an n-node DAG $G = (V = \{v_1, \ldots, v_n\}, E)$. $v_n \in V$ is the unique node with $\text{outdeg}(v_n) = 0$, which we denote the *result node*. Each node $v_i \in V$ contains the description of a polynomially-sized quantum circuit Q_i with designated input and proof registers of polynomial size (akin to a QMA-verifier). The input register's size equals $\text{indeg}(v_i)$. Q_i implicitly defines a promise problem Π^i (Π^i_{yes} contains the outputs of previous queries for which there exists a proof for Q_i that is accepted with probability $\geq 2/3$). By padding Π^i with 0^n, we get $\Pi^i \in \text{QMA}$ (v_i might have very few inputs but Q_i may require a proof of size $\text{poly}(n)$).

We say $G \in \text{QMA-DAG}_{\text{yes}}$ if the verification procedure (see Algorithm 3.3) outputs 1 *deterministically* (i.e., $\textsc{Verify}(G) = 1$), and $G \in \text{QMA-DAG}_{\text{no}}$ if it outputs 0 deterministically. We call a string $x \in \{0,1\}^n$ that can be constructed by this procedure a *correct query string*.

Algorithm 3.3 Verification procedure for QMA-DAG

function $\textsc{Verify}(G = (V, E))$
  Sort the nodes of V topologically into $v_1, \ldots, v_n$.
  Denote by $x_i \in \{0,1\}$ the result of v_i's query.
  for $i = 1, \ldots, n$ **do**
    $z_i \leftarrow \bigcirc_{v_j \in \text{pred}(v_i)} x_j$ ▷ The concatenation order is specified by C_i.
    $x_i \leftarrow \begin{cases} 1, & \text{if } z_i \in \Pi^i_{\text{yes}} \\ 0, & \text{if } z_i \in \Pi^i_{\text{no}} \\ 0 \text{ or } 1 \text{ (nondeterministically)}, & \text{if } z_i \in \Pi^i_{\text{inv}} \end{cases}$
  return x_n

Note that the correct query string is not necessarily unique anymore (compared to NP-DAG) because the graph may contain invalid queries.

Lemma 3.20. QMA-DAG *is* P^{QMA}*-complete.*

Proof. The proof is analogous to Lemma 3.2, since a quantum circuit can perform the required classical computations prior to computing the result of the actual query just as well as a classical circuit. We remark that invalid queries are no issue since $\text{P}^{\text{QMA[log]}}$ must by definition accept yes-instances and reject no-instances, regardless of how invalid queries are answered (see Definition 2.11). □

We define TREES(QMA) analogously to Definition 3.3. We want to prove the following Theorem (analogue of Theorem 3.4).

Theorem 3.21. $\text{TREES(QMA)} = \text{P}^{\text{QMA[log]}}$.

The first direction is again straightforward.

Lemma 3.22. $\text{TREES(QMA)} \supseteq \text{P}^{\text{QMA[log]}}$.

Proof. Same as Lemma 3.5. □

It remains to show that

Lemma 3.23. $\text{TREES(QMA)} \subseteq \text{P}^{\text{QMA[log]}}$.

For this we will present two proofs. The first reduces the problem to APX-SIM (see Section 3.3), and the second constructs a total solution weight function similar to Gottlob's proof (see Section 3.4).

3.3 Proof with a Query Hamiltonian

The goal of this section is to prove Lemma 3.23. We will do so by reducing TREES(QMA) to APX-SIM, which is contained in $\mathrm{P}^{\mathrm{QMA[log]}}$.

The basic idea of this proof is based on Ambainis' *query Hamiltonian* construction, which is used to prove that APX-SIM is $\mathrm{P}^{\mathrm{QMA[log]}}$-hard [4]. We present the construction here for comparison without proof of correctness.

3.3.1 Ambainis' Query Hamiltonian

Let M be a $\mathrm{P}^{\mathrm{QMA[log]}}$ machine. We assume M's oracle queries are in the form of k-LH queries with $\alpha = \varepsilon$ and $\beta = 3\varepsilon$ for some $\varepsilon \geq 1/\operatorname{poly}(n)$. Without loss of generality, M always asks $m = O(\log n)$ queries, where n is the input length. Let $H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}} \in \operatorname{Herm}(\mathcal{Y}_i)$ be the Hamiltonian that M asks if the previous $i-1$ queries have been answered with $y_1, \ldots, y_{i-1}$. The query Hamiltonian will act on the larger space $\mathcal{X} \otimes \mathcal{Y}$, where $\mathcal{X} = \bigotimes_{i=1}^m \mathcal{X}_i = \mathcal{B}^{\otimes m}$ and $\mathcal{Y} = \bigotimes_{i=1}^m \mathcal{Y}_i$. $\mathcal{X}$ intends to encode the answer of query i and $\mathcal{Y}_i$ the ground state of the corresponding Hamiltonian $H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}}$. We define the query Hamiltonian

$$H = \sum_{i=1}^{m} \frac{1}{4^{i-1}} \sum_{y_1,\ldots,y_{i-1}} \bigotimes_{j=1}^{i-1} |y_j\rangle\langle y_j|_{\mathcal{X}_j} \otimes \left(2\varepsilon|0\rangle\langle 0|_{\mathcal{X}_i} \otimes I_{\mathcal{Y}_i} + |1\rangle\langle 1|_{\mathcal{X}_i} \otimes H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}}\right).$$

We assume identities on the unspecified subspaces.

The intuition behind the construction is that if $\lambda_{\min}(H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}}) \leq \varepsilon$ (i.e., the query is answered positively), then it is energetically better to "select" the $|1\rangle\langle 1|_{\mathcal{X}_i} \otimes H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}}$ term and therefore $\mathcal{X}_i$ will be set to $|1\rangle$. On the other hand, if $\lambda_{\min}(H_{\mathcal{Y}_i}^{i,y_1\ldots y_{i-1}}) \geq 3\varepsilon$, then it is better to select the $2\varepsilon|0\rangle\langle 0|_{\mathcal{X}_i} \otimes I_{\mathcal{Y}_i}$ term by setting $\mathcal{X}_i$ to $|0\rangle$. The factor $1/4^{i-1}$ works similarly to the weighting functions from Section 3.1 by giving more weight to earlier queries.

For a formal proof, we also refer to Gharibian and Yirka [26] since the proof given in [4] appears to assume that no invalid queries are made.

3.3.2 Query Hamiltonian for TREES(QMA)

We prove a slightly more general result, akin to Theorem 3.11.

Theorem 3.24. *Let $G = (V, E)$ be an n-node* QMA-DAG *instance and ν a 2-admissable polynomial-time weighting function. If $W_\nu(G) \leq \operatorname{poly}(n)$, then the question whether $G \in$ QMA-DAG is reducible to* APX-SIM *($k = 6$ for locality).*

Let G be defined as above. G can potentially ask an exponential number of different queries. Therefore, we cannot use a Hamiltonian with a term for each possible query as above. Instead, we will make use of the fact that each node in the query tree defines a promise problem $\Pi^i \in$ QMA and provides a verifier Q_i. We will construct Hamiltonians H^i from each Q_i using a modified version of the construction from Section 2.3.2.1.

H^i acts on the space $\mathcal{Y}_i = \mathcal{X}^i \otimes \mathcal{Z}_i$, where $\mathcal{X}^i = \bigotimes_{v_j \in \operatorname{pred}(v_i)} \mathcal{X}_j$ represents the answers of inputs to Q_i, $\mathcal{X}_j = \mathcal{B}$ the answer to oracle query Q_j, and $\mathcal{Z}_i = \mathcal{B}^{\otimes m}$ is an ancillary space only H_i operates on. We define the 5-local Hamiltonian

$$H_{\mathcal{Y}_i}^i = H'_{\mathrm{in}} + H_{\mathrm{prop}} + H_{\mathrm{out}} + H_{\mathrm{stab}}$$

by applying the construction from Section 2.3 to Q_i. H'_{in} is defined by removing the constraints on the input register A from H_{in}:

$$H'_{\text{in}} = \sum_{j=1}^{m} I_{AB} \otimes \left(\Pi_j^{(0)}\right)_C \otimes |0\rangle\langle 0|_D.$$

Register A corresponds to space $\mathcal{X}^i$ and registers B, C, D to $\mathcal{Z}_i$.

Lemma 3.25. *We can construct H^i such that it has an orthonormal eigenbasis of the form*

$$\left\{|x\rangle|\lambda^i_{x,j}\rangle\right\}_{\substack{x\in\{0,1\}^{n_i} \\ j\in[2^m]}}, \tag{3.6}$$

where n_i is the number of qubits in $\mathcal{X}^i$.

Proof. We may assume that $Q_i = V_\ell \cdots V_1$ copies its input to the ancillary space using CNOT gates. Therefore, it holds for all $t \in [\ell]$, $x \in \{0,1\}^{n_i}$ and $|\phi\rangle \in \mathcal{Z}_i$ that

$$V_t|x\rangle|\phi\rangle = |x\rangle|\phi'\rangle$$

for some $|\phi'\rangle \in \mathcal{Z}_i$. Of course, this also holds for $V_t^\dagger$.

An analogous statement is true for H^i since the only nonidentity terms it applies on register A are the V_t and $V_t^\dagger$ terms in H_{prop}. Thus, it holds for all $x \in \{0,1\}^{n_i}$, $|\phi\rangle \in \mathcal{Z}_i$ that

$$H^i|x\rangle|\phi\rangle = \alpha|x\rangle|\phi'\rangle,$$

for some $|\phi'\rangle \in \mathcal{Z}_i$ and $\alpha \in \mathbb{R}$. Due to linearity, there exists an $H^i_x \in \text{Herm}(\mathcal{Z}_i)$ such that $|\phi'\rangle = H^i_x|\phi\rangle$. Let $\{|\lambda^i_{x,j}\rangle\}_{j\in[2^m]}$ be an orthonormal eigenbasis of H^i_x. Then (3.6) is clearly an orthonormal eigenbasis of H^i. □

Lemma 3.26. *We can construct H^i with the following properties for some $\varepsilon \geq 1/\operatorname{poly}(n)$:*

(1) For all $x \in \Pi^i_{\text{yes}}$, there exists a state $|\psi\rangle \in |x\rangle \otimes \mathcal{Z}_i$, such that $\langle\psi|H^i|\psi\rangle \leq \varepsilon$.

(2) For all $x \in \Pi^i_{\text{no}}$ and states $|\psi\rangle \in |x\rangle \otimes \mathcal{Z}_i$, it holds that $\langle\psi|H^i|\psi\rangle \geq 3\varepsilon$.

Proof. Let $x \in \Pi^i_{\text{yes}} \cup \Pi^i_{\text{no}}$ and define $H^i_x = H_{\text{in}} + H_{\text{prop}} + H_{\text{out}} + H_{\text{stab}}$ analogously to H^i but using the unmodified

$$H_{\text{in}} = \left(\sum_{j=1}^{n_i} \left(\Pi_j^{(\overline{x_j})}\right)_A \otimes I_{BC} + \sum_{j=1}^{m} I_{AB} \otimes \left(\Pi_j^{(0)}\right)_C\right) \otimes |0\rangle\langle 0|_D.$$

Choosing appropriate completeness/soundness parameters for Q_i, it follows from Lemma 2.32 that $\langle\eta|H^i_x|\eta\rangle \leq \varepsilon$, where $|\eta\rangle$ is the history state from (2.2). Due to Lemma 3.25, $|\eta\rangle$ has the form $|x\rangle|\eta'\rangle$, which proves (1).

For $x \in \Pi_{\text{no}}$, Lemma 2.33 implies that $\lambda_{\min}(H^i_x) \geq 3\varepsilon$. (2) follows from the fact that $H^i_x|\psi\rangle = H^i|\psi\rangle$ for all $|\psi\rangle \in |x\rangle \otimes \mathcal{Z}_i$. □

Using the H^i, we define the query Hamiltonian. Let ν be a 2-admissable weighting function.

$$\begin{aligned} H &= \sum_{i=1}^{n} \nu(v_i) \left(2\varepsilon|0\rangle\langle 0|_{\mathcal{X}_i} \otimes I_{\mathcal{Y}_i} + |1\rangle\langle 1|_{\mathcal{X}_i} \otimes H^i_{\mathcal{Y}_i}\right) \\ &=: \sum_{i=1}^{n} \nu(v_i) M_i \end{aligned}$$

H acts on the space $\mathcal{X} \otimes \mathcal{Z}$, where $\mathcal{Z} = \bigotimes_{i=1}^{n} \mathcal{Z}_i$ and $\mathcal{X} = \bigotimes_{i=1}^{n} \mathcal{X}_i$. H is 6-local since the H^i are 5-local. Note that H does not strictly fit Definition 2.28 due to the polynomially sized weights $\nu(v_i)$. We can remedy that by rescaling H with $1/W_\nu(G)$, which is not an issue since $W_\nu(G) \leq \mathrm{poly}(n)$.

Lemma 3.27. *H has an orthonormal eigenbasis of the form*

$$\left\{ |x\rangle_{\mathcal{X}} \otimes \bigotimes_{i=1}^{n} |\lambda^i_{x,j_i}\rangle_{\mathcal{Z}_i} \right\}_{\substack{x \in \{0,1\}^n \\ j_1,\ldots,j_n \in [2^m]}}. \tag{3.7}$$

Proof. This follows from Lemma 3.25 since the summands of H commute and the proof spaces $\mathcal{Z}_i$ of the H^i are distinct. □

The next lemma shows that the ground state of H must contain a correct query string.

Lemma 3.28. *Let $\lambda = \lambda_{\min}(H)$. For all incorrect query strings $x \in \{0,1\}^n$ and $|\psi\rangle \in \mathcal{Z}$, it holds that $\langle\phi|H|\phi\rangle \geq \lambda + \varepsilon$, where $|\phi\rangle = |x\rangle|\psi\rangle$.*

Proof. Let $x \in \{0,1\}^n$. Due to Lemma 3.27, the expression $\langle\phi|H|\phi\rangle$ is minimized by some $|\psi\rangle = |x\rangle_{\mathcal{X}} \otimes \bigotimes_{i=1}^{n} |\psi_i\rangle_{\mathcal{Z}_i}$. Thus, it suffices to show the lemma for $|\psi\rangle$ of that form.

The proof is now similar to Lemma 3.10. Let $i \in [n]$ such that x_i is incorrect and define $J := \{j \in [n] \mid v_j \notin v_i\uparrow, j \neq i\}$. It then holds that

$$\left(z_i \in \Pi^i_{\text{yes}} \ \wedge \ x_i = 0\right) \ \vee \ \left(z_i \in \Pi^i_{\text{no}} \ \wedge \ x_i = 1\right) \tag{3.8}$$

where z_i is defined as in Definition 3.19.

Let $|\phi'\rangle \in \mathcal{H}$ minimizing $\langle\phi'|H|\phi'\rangle$ with

$$\mathcal{H} := \bigotimes_{j \in J} \left(|x_j\rangle_{\mathcal{X}_j} |\psi_j\rangle_{\mathcal{Z}_j}\right) \otimes |\overline{x_i}\rangle_{\mathcal{X}_i} \otimes \mathcal{Z}_i \otimes \bigotimes_{v_j \in v_i\uparrow} (\mathcal{X}_j \otimes \mathcal{Z}_j).$$

We may assume that $|\phi'\rangle$ has the form

$$\begin{aligned} |\phi'\rangle &= \bigotimes_{j \in J} \left(|x_j\rangle_{\mathcal{X}_j} |\psi_j\rangle_{\mathcal{Z}_j}\right) \otimes |\overline{x_i}\rangle_{\mathcal{X}_i} |\psi'_i\rangle_{\mathcal{Z}_i} \otimes \bigotimes_{v_j \in v_i\uparrow} \left(|x'_j\rangle_{\mathcal{X}_j} |\psi'_j\rangle_{\mathcal{Z}_j}\right) \\ &=: |x\rangle_{\mathcal{X}} |\psi'_1\rangle_{\mathcal{Z}_1} \cdots |\psi'_n\rangle_{\mathcal{Z}_n} \end{aligned}$$

due to Lemma 3.27. It holds for all $j \in J$ that $x_j = x'_j$, and $|\psi'_j\rangle = |\psi_j\rangle$. Thus, $z_j = z'_j$ for all $j \in J \cup \{i\}$.

It holds for all $j \in [n]$,

$$\begin{aligned} \langle\phi|M_j|\phi\rangle &= (1 - x_j) \cdot 2\varepsilon + x_j \cdot \langle z_j|\langle\psi_j|H^i|z_j\rangle|\psi_j\rangle \\ \langle\phi'|M_j|\phi'\rangle &= (1 - x'_j) \cdot 2\varepsilon + x'_j \cdot \langle z'_j|\langle\psi'_j|H^i|z'_j\rangle|\psi'_j\rangle \end{aligned}$$

Thus, for $j \in J$

$$\langle\phi|M_j|\phi\rangle = \langle\phi'|M_j|\phi'\rangle. \tag{3.9}$$

Since x_i is incorrect, one of the two cases in (3.8) is true. In both cases,

$$\langle\phi|M_j|\phi\rangle - \langle\phi'|M_j|\phi'\rangle \geq \varepsilon. \tag{3.10}$$

For $x_i = 0$, we have $\langle\phi|M_j|\phi\rangle = 2\varepsilon$ and $\langle\phi'|M_j|\phi'\rangle \leq \varepsilon$ due to Lemma 3.26 (1). For $x_i = 1$, we have $\langle\phi|M_j|\phi\rangle \geq 3\varepsilon$ due to Lemma 3.26 (2) and $\langle\phi'|M_j|\phi'\rangle = 2\varepsilon$. Note that both statements make use of the fact that $|\phi\rangle$ was chosen to minimize $\langle\phi'|H|\phi'\rangle$.

Again due to the choice of $|\phi'\rangle$, we have for all $v_j \in v_i\uparrow$,

$$\langle\phi|M_j|\phi\rangle \leq 2\varepsilon. \tag{3.11}$$

Hence,

$$\begin{aligned}
\langle\phi|H|\phi\rangle - \langle\phi'|H|\phi'\rangle &= \sum_{j=1}^{n} \nu(v_j)\left(\langle\phi|M_j|\phi\rangle - \langle\phi'|M_j|\phi'\rangle\right) \\
&\geq \nu(v_i)\cdot\varepsilon - \sum_{v_j\in v_i\uparrow} \nu(v_j)\cdot 2\varepsilon && \text{(apply (3.9),(3.10),(3.11))} \\
&\geq \nu(v_i)\cdot\varepsilon - (\nu(v_i)-1)\cdot\varepsilon && (\nu \text{ is 2-admissable}) \\
&= \varepsilon.
\end{aligned}$$

Thus, $\langle\phi|H|\phi\rangle \geq \langle\phi'|H|\phi'\rangle + \varepsilon \geq \lambda + \varepsilon$, which concludes the proof. □

We are finally ready to prove the theorem.

Proof of Theorem 3.24. Given G, we construct H as defined above. Let $\lambda = \lambda_{\min}(H)$. The observable $A = |0\rangle\langle 0|_{\mathcal{X}_n}$ then measures the output of the result node of G. We choose as completeness/soundness thresholds $\alpha = 0, \beta = 1/2$ and $\delta = \varepsilon/2$.

Completeness: By Lemma 3.28, there exists a ground state $|\phi\rangle = |x\rangle_{\mathcal{X}}|\psi\rangle_{\mathcal{Z}}$, where x is a correct query string. For $G \in \text{QMA-DAG}_{\text{yes}}$, we therefore have $x_n = 1$ and thus $\langle\phi|A|\phi\rangle = 0$.

Soundness: Let $G \in \text{QMA-DAG}_{\text{no}}$ and $|\phi\rangle \in \mathcal{X}\otimes\mathcal{Z}$ such that $\langle\phi|H|\phi\rangle \leq \lambda+\delta$. By Lemma 3.27, we can write $|\phi\rangle = \sqrt{\gamma}|\phi^+\rangle + \sqrt{1-\gamma}|\phi^-\rangle$ for some $\gamma \in [0,1]$, such that $|\phi^+\rangle$ is a superposition of eigenvectors with correct query string and $|\phi^-\rangle$ is a superposition of eigenvectors with incorrect query string. $|\phi^+\rangle$ and $|\phi^-\rangle$ are orthogonal. Trivially, we have $\langle\phi^+|H|\phi^+\rangle \geq \lambda$. Due to Lemma 3.28, we have $\langle\phi^-|H|\phi^-\rangle \geq \lambda + \varepsilon$. Therefore,

$$\begin{aligned}
&\gamma\lambda + (1-\gamma)(\lambda+\varepsilon) \leq \langle\phi|H|\phi\rangle \leq \lambda + \varepsilon \\
\Rightarrow\quad & -\gamma\varepsilon \leq \varepsilon - \delta \\
\Rightarrow\quad & \gamma \geq \frac{\varepsilon-\delta}{\varepsilon} = \frac{1}{2}.
\end{aligned}$$

Since $G \in \text{QMA-DAG}_{\text{no}}$, we have $\langle\phi^+|A|\phi^+\rangle = 1$. As (3.6) is also an eigenbasis of A, we have

$$\langle\phi|A|\phi\rangle = \gamma\langle\phi^+|A|\phi^+\rangle + (1-\gamma)\langle\phi^-|A|\phi^-\rangle \geq \gamma \geq \frac{1}{2}.$$

□

3.3.3 Flattening the Tree

To be able to apply Theorem 3.24, we show in the following lemma that all n-node QMA-DAG instances G, where G is a tree, can be converted to an equivalent G', such that $W_\omega(G) \leq \text{poly}(n)$. ω is the 2-admissable weighting function from Lemma 3.8.

Lemma 3.29. *Let G be a* QMA-DAG *instance, where G is a tree. We can compute an equivalent G' in polynomial time (i.e., $G' \in \text{QMA-DAG}_{\text{yes}}$ if $G \in \text{QMA-DAG}_{\text{yes}}$ and $G' \in \text{QMA-DAG}_{\text{no}}$ if $G \in \text{QMA-DAG}_{\text{no}}$), such that $W_\omega(G) \leq \text{poly}(n)$.*

Proof. The proof is the same as Lemma 3.17 since the hypertree construction works for QMA-DAG in the same way. One thing we need to be careful about with regards to promise problems is creating copies of circuits during transformations of the hypertree. The potential issue is that multiple copies of the same invalid query may be answered differently. However, as argued in Lemma 3.17, we only create copies of queries during fusions and only once for each query. The output circuit then passes the output of one of those copies through. Therefore, only one copy of each query contributes to the output of the result node. □

Lemma 3.23 follows due to Theorem 3.24,

3.4 Proof with Total Solution Weight Function

In this section, we present an alternative proof to Lemma 3.23 using a total solution weight function similar to (3.5). The lemma is implied by the following theorem.

Theorem 3.30. *Let ν be a 5-admissable polynomial-time computable weighting function (see Definition 3.6). A* QMA-DAG *instance $G = (V = \{v_1, \ldots, v_n\}, E)$ with $W_\nu(G) \leq \text{poly}(n)$ can be decided in polynomial time with $O(\log n)$* QMA*-oracle queries.*

Assume each circuit Q_i receives a proof of size $m \leq \text{poly}(n)$. Define

$$t : \{0,1\}^n \times \left(\mathcal{B}^{\otimes m}\right)^n \to \mathbb{R},$$
$$t(x, \psi_1, \ldots, \psi_n) = \sum_{i=1}^{n} \nu(v_i) \left([\![Q_i(z_i, \psi_i) = x_i]\!] + \frac{x_i}{2} \right), \tag{3.12}$$

where z_i is defined as in Definition 3.19 and $Q_i(z_i, \psi_i)$ is the random variable corresponding to Q_i's output given input z_i and proof $|\psi_i\rangle$. $[\![X = b]\!]$ is defined as the random variable that is 1 if $X = b$ and 0 if $X \neq b$. Comparing (3.5) and (3.12), the main difference is that the latter is a random variable. Moreover, (3.12) uses the $x_i/2$ terms because we can no longer directly check if the x_i values match the circuit outputs due to invalid queries.

We define

$$T := \max_{x, |\psi_1\rangle, \ldots, |\psi_n\rangle} \mathbb{E}[t(x, \psi_1, \ldots, \psi_n)].$$

Note that the value of T does not change if we allow mixed states as proofs. For $\varepsilon > 0$, define the promise problem Π with

$$\Pi_{\text{yes}} = \{(t, s) \mid T \geq s\}$$
$$\Pi_{\text{no}} = \{(t, s) \mid T \leq s - \varepsilon\}$$

Lemma 3.31. *For $\varepsilon \geq 1/\operatorname{poly}(n)$, $\Pi \in \mathrm{QMA}$.*

Proof. The idea is to sample t multiple times and then apply Hoeffding's inequality (Lemma 2.1). Specifically, we construct a QMA verifier Q that gets input t, s and receives as proof x and Nmn qubits (to be used as proofs for the Q_i circuits) for sufficiently large N (to be determined later). For each sample of t, Q evaluates all $[\![Q_i(z_i, \psi_i) = x_i]\!]$ expressions and classically computes the corresponding output of t. Let $X_1, \ldots, X_N$ denote the outcomes of these samples and $\overline{X}$ their mean. Q then accepts if $\overline{X} \geq s - \varepsilon/2$ and otherwise rejects.

Completeness: Assume it holds $T \geq s$. The prover sends a proof

$$|x\rangle \bigotimes_{i=1}^{n} |\psi_i\rangle^{\otimes N},$$

such that

$$\mathbb{E}[X_j] = \mathbb{E}[t(x, \psi_1, \ldots, \psi_n)] = T = \mathbb{E}[\overline{X}].$$

Let $W = 2 \cdot W_\nu(G) \leq \operatorname{poly}(n)$. Then each X_j is bounded by $[0, W]$. By Hoeffding's inequality, we have

$$\Pr\left[|\overline{X} - T| \geq \varepsilon/2\right] \leq 2 \exp\left(-\frac{N\varepsilon^2}{2W^2}\right) \leq \frac{1}{3},$$

choosing $N \leq \operatorname{poly}(n)$ to be sufficiently large. Since $T \geq s$, Q accepts with probability $\geq 2/3$.

Soundness: Assume $T \leq s - \varepsilon$. Now we cannot just assume that each sample of t receives the same proof. We argue that $\Pr[\overline{X} \geq T - \varepsilon/2]$ is still maximized by a separable proof. Let $\{E_i^0, E_i^1\}$ be the POVM for Q_i. Then $E_i^0 + E_i^1 = I$ and $\operatorname{Tr}(E_i^1 \rho) = \Pr[Q_i \text{ accepts } \rho]$. Note that E_i^0 and E_i^1 are simultaneously diagonalizable with eigenbasis $\{|\lambda_{i,k}\rangle\}_k$. Let $z \in \{0,1\}^{nN}$ denote the outcomes of all copies of all Q_i. Denote by t_z the value computed for $\overline{X}$, given outcomes z. Let $Z = \{z : t_z \geq T + \varepsilon/2\}$ and

$$E = \sum_{z \in Z} \bigotimes_{i,j} E_i^{z_{i,j}}.$$

Then,

$$\Pr_\rho[\overline{X} \geq T + \varepsilon/2] = \operatorname{Tr}(E\rho),$$

where $\rho \in \operatorname{Herm}(\mathcal{B}^{\otimes Nmn})$ is the entire proof given to the verifier. E then has an eigenbasis of separable states

$$B_E = \left\{ \bigotimes_{i,j} |\lambda_{i,k_{i,j}}\rangle \right\}_{(k_{i,j})_{i,j}}.$$

Hence, $\operatorname{Tr}(E\rho)$ becomes maximal for $\rho = |\psi\rangle\langle\psi|$ and some $|\psi\rangle = \bigotimes_{i,j} |\psi_{i,j}\rangle \in B_E$. Thus, it suffices to bound

$$\begin{aligned}
\operatorname{Tr}(E|\psi\rangle\langle\psi|) &= \langle\psi|E|\psi\rangle \\
&= \sum_{z \in Z} \Pr[\forall i, j : j\text{-th copy of } Q_i \text{ has outcome } z_{i,j}] \\
&= \Pr\left[\frac{1}{N} \sum_{j=1}^{N} \underbrace{t(x, \psi_{1,j}, \ldots, \psi_{n,j})}_{X_j} \geq T + \varepsilon/2\right] \\
&= \Pr[\overline{X} \geq T + \varepsilon/2].
\end{aligned}$$

Therefore, the X_j are independent with $\mathbb{E}[X_j] \leq T$ and $X_j \in [0, W]$. Thus, $\mathbb{E}[\overline{X}] \leq T$. Hence, we can apply Hoeffding's inequality again:

$$\Pr\left[|\overline{X} - \mathbb{E}[\overline{X}]| \geq \varepsilon/2\right] \leq \frac{1}{3}$$

Since $\mathbb{E}[\overline{X}] \leq T \leq s - \varepsilon$ for all k, Q accepts with probability $\leq 1/3$. □

Hence, we can determine T up to an additive error of ε using $O(\log n)$ QMA-oracle queries.

Lemma 3.32. *If $\mathbb{E}[t(x, \psi_1, \ldots, \psi_n)] > T - 1/3$, then x is a correct query string.*

Proof. This proof is very similar to Lemmas 3.10 and 3.28. It holds that

$$\mathbb{E}[t(x, \psi_1, \ldots, \psi_n)] = \sum_{i=1}^{n} \nu(v_i) \left(\Pr[Q_i(z_i, \psi_i) = x_i] + \frac{x_i}{2}\right).$$

Let x be an incorrect query string with incorrect x_i. It holds that $z_i \in \Pi^i_{\text{yes}} \cup \Pi^i_{\text{no}}$.

Define $J := \{j \in [n] \mid v_j \notin v_i\uparrow,\ j \neq i\}$. For $j \in J$, set $x'_j := x_j$ and $|\psi'_j\rangle := |\psi_j\rangle$. Set $x'_i := \overline{x_i}$ and choose $|\psi'_i\rangle$, and $|\psi'_j\rangle, x'_j$ for all $v_j \in v_i\uparrow$ to maximize

$$\sum_{v_j \in v_i\uparrow \cup \{v_i\}} \nu(v_j) \left(\Pr[Q_j(z'_j, \psi'_j) = x'_j] + \frac{x'_j}{2}\right).$$

Let $1 - \varepsilon$ and ε be the completeness/soundness parameters for all Q_i (we choose ε later). Since x_i is incorrect, we have for $x_i = 0$,

$$\begin{aligned}
\Pr[Q_i(z_i, \psi_i) = x_i] + \frac{x_i}{2} &\leq 1 \\
\Pr[Q_i(z'_i, \psi'_i) = x'_i] + \frac{x'_i}{2} &\geq 1 - \varepsilon + \frac{1}{2} = \frac{3}{2} - \varepsilon,
\end{aligned}$$

and for $x_i = 1$,

$$\begin{aligned}
\Pr[Q_i(z_i, \psi_i) = x_i] + \frac{x_i}{2} &\leq \varepsilon + \frac{1}{2} \\
\Pr[Q_i(z'_i, \psi'_i) = x'_i] + \frac{x'_i}{2} &\geq 1 - \varepsilon.
\end{aligned}$$

Hence,

$$\left(\Pr[Q_i(z'_i, \psi'_i) = x'_i] + \frac{x'_i}{2}\right) - \left(\Pr[Q_i(z_i, \psi_i) = x_i] + \frac{x_i}{2}\right) \geq \frac{1}{2} - 2\varepsilon. \tag{3.13}$$

For $\varepsilon \leq 1/12$,

$$\begin{aligned}
&t(x', \psi_1', \ldots, \psi_n') - t(x, \psi_1, \ldots, \psi_n) \\
&\quad = \sum_{j=1}^{n} \nu(v_j) \left(\Pr[Q_j(z_j', \psi_j') = x_j'] + \frac{x_j'}{2} \right) - \sum_{j=1}^{n} \nu(j) \left(\Pr[Q_j(z_j, \psi_j) = x_j] + \frac{x_j}{2} \right) \\
&\quad \geq \frac{1}{3}\nu(v_i) - \sum_{v_j \in v_i\uparrow} \nu(j) \left(\Pr[Q_j(z_j', \psi_j') = x_j'] + \frac{x_j'}{2} \right) && \text{(apply (3.13))} \\
&\quad \geq \frac{1}{3}\nu(v_i) - \frac{3}{2} \sum_{v_j \in v_i\uparrow} \nu(j) \\
&\quad = \frac{1}{3} \left(\nu(v_i) - \frac{9}{2} \sum_{v_j \in v_i\uparrow} \nu(v_j) \right) \\
&\quad > \frac{1}{3}. && (\nu \text{ is 5-admissable})
\end{aligned}$$

Thus, $T \geq t(x', \psi_1', \ldots, \psi_n') \geq t(x, \psi_1, \ldots, \psi_n) + 1/3$. □

After determining T using Lemma 3.31, we determine x_n for a correct query string using a final query. This query is almost the same as outlined in Lemma 3.31, except that it only accepts if $x_n = 1$. The correctness of this final query then follows from Lemma 3.32. This completes the proof of Theorem 3.30.

3.5 Further Complexity Classes

A natural question one can ask is whether these results can be generalized to further complexity classes. This turns out to be true since the proofs in Sections 3.3.3 and 3.4 work with only trivial modifications for MA-*like* complexity classes (i.e., there exists a verifier circuit that behaves similarly to MA). It is, however, necessary for Section 3.4 that some form of amplification is possible (specifically Lemma 3.31). For QMA(2), we can use separate proofs for all query samples since QMA(2) = QMA(poly) [33].

Theorem 3.33. *For $C \in \{\mathrm{MA}, \mathrm{QCMA}, \mathrm{QMA}, \mathrm{QMA}(2)\}$, $\mathrm{TREES}(C) = P^{C[\log]}$.*

It is not clear whether this result also holds for StoqMA, since StoqMA does not permit gap amplification (see Definition 2.13). A different idea to prove the theorem for StoqMA would be the Hamiltonian construction from Section 3.3. This would require a different circuit Hamiltonian construction specifically for StoqMA. It seems the one given by Bravyi, Bessen and Terhal [12] might be suitable. There exists a $\mathrm{P}^{\mathrm{StoqMA}[\log]}$-complete variant of APX-SIM [24].

3.6 Bounded Treewidth Dependency Graph

In this section, we return to the classical setting and extend Gottlob's result (see Section 3.1) to the case where the query dependency graph has bounded treewidth (see Definition 2.2).

Definition 3.34 (BTW(NP))**.** A language L is in BTW(NP) (bounded treewidth) if there exists a constant k such that L is reducible to an NP-DAG instance G with $\mathrm{tw}(G) \leq k$.

We will prove the following theorem.

Theorem 3.35. $\mathrm{BTW}(\mathrm{NP}) = \mathrm{P}^{\mathrm{NP}[\log]}$.

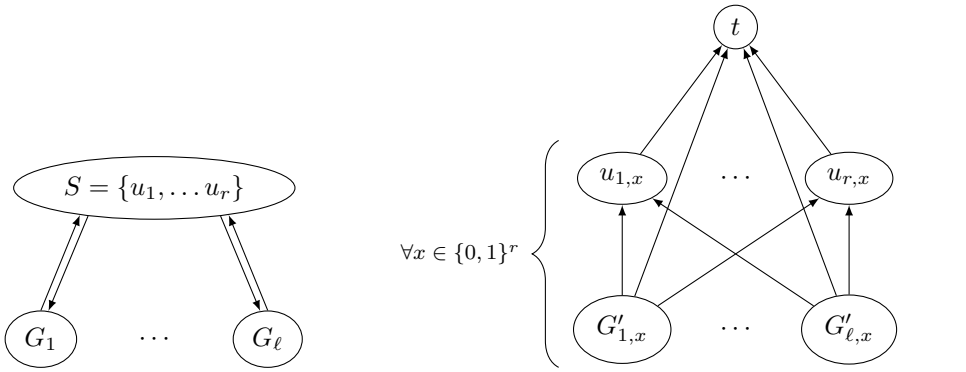
Trivially, $\mathrm{BTW}(\mathrm{NP}) \supseteq \mathrm{TREES}(\mathrm{NP}) = \mathrm{P}^{\mathrm{NP}[\log]}$. Thus, it only remains to show $\mathrm{BTW}(\mathrm{NP}) \subseteq \mathrm{P}^{\mathrm{NP}[\log]}$. For that, we introduce the notion of *separator trees* that will allow us to hierarchically decompose graphs with bounded treewidth.

Definition 3.36. A *separator tree* of an undirected graph $G = (V, E)$ is a tree $T = (\mathcal{S} = \{S_1, \ldots, S_m\}, E_T)$ rooted in S_1 such that $\bigcup_{i=1}^{m} S_i = V$, S_1 is a balanced separator of G, and the trees rooted in the children of S_1 are separator trees of $G \setminus S_1$.

Lemma 3.37. *Let $G = (V, E)$ be a graph with $\mathrm{tw}(G) \leq k$. A separator tree with maximum separator size k can be computed in polynomial time.*

Proof. By Lemma 2.4, we have $\mathrm{s}(G) \leq k$. Therefore, every induced subgraph of G has a balanced separator. Thus, we can construct the separator tree by finding a balanced separator S of G with $|S| \leq k$ and then recursing on the connected components of $G \setminus S$. □

Theorem 3.38. *Fix a constant $s > 0$. Let G be an* NP-DAG *instance with $\mathrm{s}(G) \leq s$. There exists a polynomial-time algorithm that computes an equivalent* NP-DAG *instance G' such that $|v\uparrow| = O(\log n)$ for all $v \in V(G')$.*



(a) Original graph G.

(b) Graph G' constructed in Theorem 3.38.

Figure 3.2: Illustration of Algorithm 3.4.

Proof. The basic idea is to split the query graph using a separator $S = \{u_1, \ldots, u_r\}$ into components $G_1, \ldots, G_\ell$. Note that both nodes in S as well as in G_i may have inputs from S. Therefore, for all possible outputs $x \in \{0,1\}^r$ of queries in S, we create copies $G_{i,x}$ of G_i and $u_{i,x}$ of u_i in which all inputs from S are hardcoded to x. Inputs from G_i in $u_{i,x}$ are computed from $G_{i,x}$. We recurse on the $G_{i,x}$ to transform them into $G'_{i,x}$. This algorithm is formally given in Algorithm 3.4. Note that the returned graph does not have a result node. Instead it contains many copies of the original one. We add a new result node t that selects the correct copy of the original result node using GetOutput. This construction is illustrated in Figure 3.2.

Formally, we compute a separator tree $T = (\mathcal{S}, E_T)$ of G (using Lemma 3.37). Then, we compute $G'' = \textsc{Squeeze}(G, T)$. Let v be the result node of G. We obtain G' by adding a node t to G'' with inputs from all other nodes in G'' that computes the result using $\textsc{GetOutput}(v, T, G'', y)$ as in Line 20 of Squeeze (y is constructed from the inputs of t).

Algorithm 3.4 Transform a query graph G with $\mathrm{s}(G) \leq s$ into an equivalent G' where each node has logarithmically many successors.

function SQUEEZE($G = (V, E), T = (\mathcal{S}, E_T)$)
  Let $S = \{u_1, \ldots, u_r\}$ be the root of T. ▷ $r \leq s$
  Let $T_1, \ldots, T_\ell$ be the trees rooted at the children $S_1, \ldots, S_\ell$ of S. ▷ ℓ may be 0
  Let $G_1, \ldots, G_\ell$ be the corresponding induced subgraphs of G.
  Let $G' = (V', E')$ be an empty graph.
  for all $x \in \{0,1\}^r$ **do**
    for $i = 1, \ldots, \ell$ **do**
      Let $G_{i,x}$ be a copy of G_i.
      for $j = 1, \ldots, r$ **do**
        Replace all inputs from u_j in $G_{i,x}$ with a constant x_j.
      $G'_{i,x} \leftarrow$ SQUEEZE($G_{i,x}, T_i$)
      Add $G'_{i,x}$ to G'.
    for $i = 1, \ldots, r$ **do**
      Let $u_{i,x}$ be a copy of u_i.
      for all $v \in \mathrm{pred}(u)$ **do**
        if $\exists k : v = u_k$ **then**
          Replace v with a constant x_k.
        else
          Let k such that $v \in G_k$.
          Replace v with GETOUTPUT($v, T_k, G'_{k,x}, y$). ▷ $u_{i,x}$ uses its inputs as y.
      Add $u_{i,x}$ to G' with incoming edges from all nodes in $G'_{1,x}, \ldots, G'_{\ell,x}$.
  return G'.

function GETOUTPUT($v, T, G', y : V(G') \to \{0,1\}$)
  Let $S = \{u_1, \ldots, u_r\}$ be lexicographically ordered. ▷ Reuse terminology from SQUEEZE
  $x \leftarrow 0^r$
  for $i = 1, \ldots, r$ **do**
    $x_i \leftarrow y(u_{i,x})$
  if $\exists i : v = u_i$ **then**
    return x_i
  else
    Let i such that $v \in G_i$.
    return GETOUTPUT($v, G'_{i,x}, T_i, y$)

SQUEEZE clearly has polynomial runtime excluding the recursion. T has height $O(\log n)$ and each call does at most $2^s = O(1)$ recursive calls on children. Hence, there will be at most $2^{s \cdot O(\log n)}$ calls in total, which is polynomial.

Each call of SQUEEZE adds at most 2^s outputs to nodes in $G'_{1,x}, \ldots, G'_{\ell,x}$. Hence, each node has $O(\log n)$ outputs. By construction, each node $v \in V(G')$ has direct edges to all nodes in $v{\uparrow}$. We finally show correctness. For that we need to argue that, given correct outputs y of G',

(1) $y(u_{i,x})$ is the correct output of u_i if x_j is the correct output of u_j for all $j \in [r]$,

(2) GETOUTPUT(v, T, G', y) returns the correct output for all $v \in V$.

We do this using induction on $|\mathcal{S}|$.

For $|\mathcal{S}| = 1$, G' contains copies of all queries with all possible inputs as singular nodes. Therefore, (1) follows immediately. (2) follows from the fact that GETOUTPUT sorts the queries topologically. Therefore, $y(u_{i,x})$ is the correct output for u_i as x_j (for $j < i$) was already set to the correct output of u_j, and u_i has no incoming edges from u_j for $j \geq i$.

Now let $|\mathcal{S}| > 1$ and assume (1) and (2) hold for smaller $|\mathcal{S}|$. By the inductive hypothesis,

GetOutput$(v, T_k, G'_{k,x}, y)$ (see Line 20) returns the correct output for v if x_j is the correct output for u_j for all $j \in [r]$. Thus, (1) follows because all inputs to $u_{i,x}$ are computed correctly by the inductive hypothesis (2). In GetOutput, the string x is constructed correctly since the u_i are sorted topologically and (1) (same argument as the base case). If GetOutput returns in Line 29, (2) follows from the fact that x_i is computed correctly. Otherwise, it returns in Line 32, and (2) follows from the inductive hypothesis (2). □

Theorem 3.35 then follows from Theorem 3.11 and $\nu = \omega$.

There seems to be no straightforward way to generalize this result to promise problems because the outputs of multiple copies of a query may be used in the computation of the final result (see also Lemma 3.29). Hence, G and G' are no longer necessarily equivalent. Therefore, the question whether $\mathrm{P}^{\mathrm{QMA[log]}} = \mathrm{BTW(QMA)}$ remains an interesting open question.

Chapter 4

Upper Bounds for QMA

The previous chapter dealt with the complexity class $\mathrm{P}^{\mathrm{QMA[log]}}$, which can be seen as an "upper bound" of QMA since it trivially holds that $\mathrm{QMA} \subseteq \mathrm{P}^{\mathrm{QMA[log]}}$. Another upper bound is $\mathrm{A_0PP}$, which has been shown to contain QMA by Vyalyi [61]. In this chapter, we will compare the two complexity classes $\mathrm{P}^{\mathrm{QMA[log]}}$ and $\mathrm{A_0PP}$. We begin by ruling out one approach for showing $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{A_0PP}$ by proving that $\mathrm{C_{=}P} \nsubseteq \mathrm{A_0PP}$ unless $\mathrm{PH} \subseteq \mathrm{PP}$ (see Section 4.1). Afterwards, we construct oracle separations for $\mathrm{A_0PP}$ and $\mathrm{P}^{\mathrm{QMA[log]}}$ in both directions (see Sections 4.2 and 4.3). In fact, we prove somewhat stronger results from which these oracle separations follow. This gives some evidence that neither $\mathrm{A_0PP} \subseteq \mathrm{P}^{\mathrm{QMA[log]}}$ nor $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{A_0PP}$.

4.1 $\mathrm{A_0PP}$ vs. $\mathrm{C_{=}P}$

There exists a very simple proof for the result that $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{PP}$. Gharibian and Yirka [26] proved this result using a rather involved proof based on the hierarchical voting scheme used in the proof of $\mathrm{P}^{\mathrm{NP[log]}} \subseteq \mathrm{PP}$ by Beigel, Hemachandra, and Wechsung [9].

Theorem 4.1. $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{PP}$.

Proof. This proof is based on *truth table reductions*. A language A is *polynomial-time truth table reducible* to a language B, denoted $A \leq_{\mathrm{tt}} B$, if $A \in \mathrm{P}^{\|B}$. Gharibian, Piddock, and Yirka have shown $\mathrm{P}^{\mathrm{QMA[log]}} = \mathrm{P}^{\|\mathrm{QMA}}$ [24]. Fortnow and Reingold [21] have shown that PP is *closed under truth table reductions* (i.e., $\mathrm{PP} = \mathrm{P}^{\|\mathrm{PP}}$). Since $\mathrm{QMA} \subseteq \mathrm{PP}$ [45], we have

$$\mathrm{P}^{\mathrm{QMA[log]}} = \mathrm{P}^{\|\mathrm{QMA}} \subseteq \mathrm{P}^{\|\mathrm{PP}} = \mathrm{PP}.$$

We need to be careful with the containment $\mathrm{P}^{\|\mathrm{QMA}} \subseteq \mathrm{P}^{\|\mathrm{PP}}$ since QMA is a class of promise problems, whereas $\mathrm{PP} = \mathrm{P}^{\|\mathrm{PP}}$ was only shown for languages.[1] However, we can easily transform a $\mathrm{P}^{\|\mathrm{QMA}}$-machine M into a $\mathrm{P}^{\|\mathrm{PP}}$-machine M' by converting QMA queries to PP queries. M' now implicitly defines a language, which is in PP. □

Now, if $\mathrm{A_0PP}$ were closed under truth table reductions, then $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{A_0PP}$ would follow by the same argument. We will argue that this is not likely to be true.

It trivially holds that $\mathrm{coC_{=}P} \subseteq \mathrm{A_0PP}$. If $\mathrm{A_0PP}$ were closed under truth table reductions (and thereby also under complement), then $\mathrm{C_{=}P} \subseteq \mathrm{A_0PP}$ would follow.

[1] $\mathrm{P}^{\|\mathrm{PP}} = \mathrm{PP}$ likely also holds for promise problems, but we do not prove this here.

Theorem 4.2. $\mathrm{C_{=}P} \subseteq \mathrm{A_0PP}$ *implies* $\mathrm{PH} \subseteq \mathrm{PP}$.

Proof. This is proof is essentially the same as Vyalyi's proof that $\mathrm{A_0PP} = \mathrm{PP}$ implies $\mathrm{PH} \subseteq \mathrm{PP}$ [61, Theorem 2].

Assume $\mathrm{C_{=}P} \subseteq \mathrm{A_0PP}$ and let $L \in \mathrm{PH}$. We will show that then $L \in \mathrm{PP}$. By Toda's theorem (Theorem 2.24), $L \in \mathrm{P}^{\#\mathrm{P}[1]}$. Therefore, L is recognized by a deterministic polynomial-time Turing machine M querying a #P-oracle for the value $f(y)$, where $f \in \#\mathrm{P}$.

Define a function $g_M : \{0,1\}^* \times \mathbb{Z} \to \mathbb{Z}$, where $g_M(x,j)$ is defined as the output of $M(x)$, where the oracle query is answered by j. Clearly, $g_M \in \mathrm{FP}$ and $g_M(x,j) = \chi_L(x)$ if $j = f(y)$ and $g_M(x,j) \in \{0,1\}$ otherwise.

Let $L' := \{\langle y,j\rangle \mid f(y) = j\}$ and $g'(y,j) := f(y) - j$. Then, $g'(y,j) = 0$ iff $\langle y,j\rangle \in L'$. Hence, $L' \in \mathrm{C_{=}P}$.

By our assumption, $L' \in \mathrm{A_0PP}$. Hence, there exists $\tilde{g} \in \mathrm{GapP}$, such that for $n := |x|, \ell := n + q(n)$, where $2^{q(n)} - 1$ is an upper bound on $f(y)$:

$$\begin{aligned} f(y) = j \quad &\Rightarrow \quad \tilde{g}(y,j) > 2^{p(\ell)} \\ f(y) \neq j \quad &\Rightarrow \quad 0 \leq \tilde{g}(y,j) < 2^{-q(n)} 2^{p(\ell)} \end{aligned}$$

We now define our final $g \in \mathrm{GapP}$, making use of the closure properties of GapP (shown in [20]):

$$g(x) := \sum_{j \in \{0,1\}^{q(n)}} g_M(x,j) \cdot \tilde{g}(h(x),j),$$

where $h \in \mathrm{FP}$ with $h(x) = y$.

$$\begin{aligned} x \in L \quad &\Rightarrow \quad g(x) \geq g_M(x,f(y)) \cdot \tilde{g}(y,f(y)) = \chi_L(x) \cdot \tilde{g}(y,f(y)) > 2^{p(\ell)} \\ x \notin L \quad &\Rightarrow \quad g(x) < 2^{q(n)} 2^{-q(n)} 2^{p(\ell)} \leq 2^{p(\ell)} \end{aligned}$$

It follows that $L \in \mathrm{PP}$. □

4.2 coNP vs. $\mathrm{A_0PP}$

In this section, we construct an oracle A such that $\mathrm{coNP}^A \not\subseteq \mathrm{A_0PP}^A$. We will also argue that the containment $\mathrm{SBQP} \subseteq \mathrm{A_0PP}$ relativizes. Since the containment $\mathrm{coNP} \subseteq \mathrm{P}^{\mathrm{QMA}[\log]}$ trivially relativizes, we have

$$\mathrm{coNP}^A \subseteq \mathrm{P}^{\mathrm{QMA}^A[\log]} \not\subseteq \mathrm{SBQP}^A \subseteq \mathrm{A_0PP}^A.$$

We first give a proof from first principles (Sections 4.2.1 and 4.2.2). Afterwards, we give a very simple proof using query complexity (Section 4.2.3), that was pointed out to us by William Kretschmer.

4.2.1 Technical Lemmas

In the following, we show some technical lemmas we will use for the oracle separation. Here, we define $\|f\| := \max_{x \in [-1,1]} |f(x)|$ for a function $f : [-1,1] \to \mathbb{R}$.

Lemma 4.3 (Markov [49])**.** *Let p be a polynomial with $\deg(p) \leq n$. Then, $\|p'\| \leq n^2 \|p\|$.*

Lemma 4.4. *Let p be a polynomial with $\deg(p) = n$ and $|p(i)| \leq 1$ for all $i \in [2N]$. If $n^2/N < 1-\varepsilon$ for some $\varepsilon > 0$, then $|p(0)| < \varepsilon^{-1}$.*

Proof. The following proof is adapted from [55]. Let

$$\pi : [-1, 1] \to [0, 2N],\ x \mapsto (x + 1) \cdot N,$$

and $q = p \circ \pi$ ($\circ$ denotes function composition). q is a polynomial with $\deg(q) = n$. For $k \in [N]$, define

$$\xi_k := -1 + \frac{2k-1}{N}.$$

Then,

$$\pi(\xi_k) = \pi\left(-1 + \frac{2k-1}{N}\right) = 2k - 1.$$

Thus, $|q(\xi_k)| = |p(2k-1)| \leq 1$ for all $k \in [N]$. It also holds that $q(-1) = p((-1+1)N) = p(0)$.

Let $x \in [-1, 1]$. There exists a k such that $|x - \xi_k| \leq 1/N$. By the mean value theorem, there exists a $z \in [-1, 1]$, such that

$$\begin{aligned} |q(x)| &\leq |q(\xi_k)| + |\xi_k - x| \cdot |q'(z)| \\ &\leq 1 + \frac{1}{N}\|q'\|. \end{aligned}$$

Therefore,

$$\begin{aligned} \|q\| &\leq 1 + \frac{1}{N}\|q'\| \\ &\leq 1 + \frac{n^2}{N}\|q\| && \text{(apply Lemma 4.3)} \\ &< 1 + (1-\varepsilon)\|q\|. \end{aligned}$$

Hence, $\|q\| < \varepsilon^{-1}$. □

Lemma 4.5. *Let $x_0, \ldots, x_m, T \in \mathbb{R}$ and $y_k := \sum_{i=0}^{k} \binom{k}{i} x_i$ with $y_k \in [0, \varepsilon T]$ for all $k \in [N]$, where $m^2/N < 1 - \varepsilon$ for some $\varepsilon > 0$. Then $|y_0| < T$.*

Proof. We can write binomial coefficients as polynomials:

$$\binom{t}{k} = \frac{(t)_k}{k!} = \frac{t(t-1)(t-2)\cdots(t-k+1)}{k(k-1)(k-2)\cdots 2 \cdot 1}.$$

Note that this is still correct for $0 \leq t < k$ as then the numerator has a root in t. Hence,

$$f(k) := y_k = \sum_{i=0}^{k} \binom{k}{i} x_i = \sum_{i=0}^{m} \binom{k}{i} x_i$$

is a polynomial of degree m with $|f(i)| \leq \varepsilon T$ for $i = 1, \ldots, N$. Lemma 4.4 now implies $|y_0| = |f(0)| < T$. □

Lemma 4.6. *Let*

(1) S be a set with $|S| = N$,

(2) $g : 2^S \to \mathbb{R}$ with $g(B) = 0$ for $|B| > m$, where $m^2/N < 1 - \varepsilon$ for some $\varepsilon > 0$,

(3) $f(A) = \sum_{B \subseteq A} g(B)$, $f(A) \in [0, \varepsilon T]$ *for some* $T > 0$ *and all* $\emptyset \neq A \subseteq S$.
Then $|f(\emptyset)| < T$.

Proof. Assume $|f(\emptyset)| \geq T$. We obtain a contradiction by defining the x_i for $i = 0, \ldots, N$ in Lemma 4.5 as the average of $g(B)$ over all B of size i. Thus, we set

$$x_i := \frac{1}{\binom{N}{i}} \sum_{\substack{B \subseteq S \\ |B|=i}} g(B).$$

Note that $x_i = 0$ for $i > m$. Hence,

$$y_k = \sum_{i=0}^{k} \binom{k}{i} x_i = \sum_{i=0}^{k} \frac{\binom{k}{i}}{\binom{N}{i}} \sum_{\substack{B \subseteq S \\ |B|=i}} g(B).$$

Next, we will show for all k

$$y_k = \frac{1}{\binom{N}{k}} \sum_{\substack{A \subseteq S \\ |A|=k}} f(A), \tag{4.1}$$

which implies $|y_0| \geq T$ and $y_k \in [0, \varepsilon T]$ for $k > 0$, contradicting Lemma 4.5. It holds that

$$\frac{1}{\binom{N}{k}} \sum_{\substack{A \subseteq S \\ |A|=k}} f(A) = \frac{1}{\binom{N}{k}} \sum_{\substack{A \subseteq S \\ |A|=k}} \sum_{B \subseteq A} g(B) = \frac{1}{\binom{N}{k}} \sum_{i=0}^{k} \sum_{\substack{B \subseteq S \\ |B|=i}} \binom{N-i}{k-i} g(B).$$

Regarding the second equality, observe that there are $\binom{N-i}{k-i}$ ways to choose a set A with $|A| = k$ that contains a set B with $|B| = i$. For (4.1) to hold, it is therefore sufficient that

$$\begin{aligned}
& \frac{\binom{k}{i}}{\binom{N}{i}} = \frac{\binom{N-i}{k-i}}{\binom{N}{k}} \\
\Leftrightarrow \quad & \binom{N}{k}\binom{k}{i} = \binom{N}{i}\binom{N-i}{k-i} \\
\Leftrightarrow \quad & \frac{N!k!}{k!(N-k)!i!(k-i)!} = \frac{N!(N-i)!}{i!(N-i)!(k-i)!(N-k)!} \\
\Leftrightarrow \quad & \frac{N!}{(N-k)!i!(k-i)!} = \frac{N!}{i!(k-i)!(N-k)!}.
\end{aligned}$$

The last equality holds trivially. □

4.2.2 Oracle Separation

Theorem 4.7. *There exists an oracle* A *such that* $\mathrm{coNP}^A \not\subseteq \mathrm{A_0PP}^A$.

Proof. The proof is similar to the well known oracle separation of P and NP. We define the unary language $L_A := \{1^n \mid \nexists y \in A : |y| = n\}$.

First, we show $L_A \in \mathrm{coNP}^A$ for all A. Let M be the NTM that on input $x = 1^n$ chooses a y with $|y| = n$ nondeterministically and accepts iff $y \notin A$. If $x \in L_A$, then all paths of M accept. Otherwise, there must be some rejecting path. Hence, M accepts L_A.

Next, we construct A such that $L_A \notin \mathrm{A_0PP}^A$ using diagonalization. Recall the definition of $\mathrm{A_0PP}^A$: $L \in \mathrm{A_0PP}$ iff there exist a CM M^A and $T \in \mathrm{FP}$ such that

(a) if $x \in L$, then $\mathrm{gap}_{M^A}(x) > T(x)$,
(b) if $x \notin L$, then $0 \leq \mathrm{gap}_{M^A}(x) < \varepsilon T(x)$.

We will enumerate all pairs (M_i^A, T_i) and construct $A_i \subseteq \{0,1\}^{n_i}$ such that (M_i^A, T_i) either violates the above bounds or makes a mistake on input 1^{n_i}, where we define $A = \bigcup_i A_i$ and choose n_i to be sufficiently large to satisfy the below conditions:

- For $j < i$, $M_j^A(1^{n_j})$ only asks oracle queries y with $|y| < n_i$. Hence, $\mathrm{gap}_{M_i^A}(1^{n_i}) = \mathrm{gap}_{M_i^{A^i}}(1^{n_i})$ for $A^i := \bigcup_{j=1}^{i} A_i$.
- Each execution path of $M_i^A(1^{n_i})$ asks at most m queries, where $m^2/2^{n_i/2} < 1 - \varepsilon$.

Both conditions can easily be fulfilled since the M_i^A run in polynomial time.

We can write $\mathrm{gap}_{M_i^{A^{i-1} \cup A_i}}(1^{n_i})$ as a function in A_i, i.e.,

$$f(A_i) := \mathrm{gap}_{M_i^{A^{i-1} \cup A_i}}(1^{n_i}).$$

Now consider the execution of M_i^A. We can essentially view it as a tree in which each interior node has two children and corresponds either to an oracle query or the nondeterministic choice of a single bit. Each leaf v is labeled with $\ell(v) \in \{-1, +1\}$, depending on whether the machine accepts or rejects. We can remove nodes corresponding to oracle queries not having length n_i since their outcome is fixed already (we are only interested in the dependency on A_i). Let r be the root node. For an interior node v, denote by $v_i, i \in \{0,1\}$ the child of v corresponding to oracle answer i or nondeterministic choice of bit i. Let x_v be the query string of an oracle query node v. Then $f(A_i) = h(r)$ for the following recursively defined function h:

$$h(v) = \begin{cases} \ell(v), & v \text{ is a leaf} \\ h(v_0) + h(v_1), & v \text{ is a nondeterministic bit choice} \\ (1 - A_i(x_v))h(v_0) + A_i(x_v)h(v_1), & v \text{ is an oracle query} \end{cases}$$

Thus, $f(A_i)$ has the following form, which is obtained by applying distributivity:

$$f(A_i) = h(r) = \sum_{B \subseteq \{0,1\}^{n_i}} g(B) \prod_{x \in B} A_i(x) = \sum_{B \subseteq A_i} g(B),$$

where $g(B)$ is implicitly defined by h and the resulting coefficients after expanding all terms as above. We may assume $g(B) = 0$ if B does not correspond to the set of queries actually asked along some path in the tree. Notably, $g(B) = 0$ for $|B| > m$, as M_i^A cannot make that many queries.

$S = \{0,1\}^{n_i}$ and g satisfy conditions (1) and (2) of Lemma 4.6 for $T := T_i(1^{n_i})$. Therefore, one of the following cases holds.

1. f satisfies (3). Therefore, $f(\emptyset) < T$. This means that either $M_i^{A^{i-1}}$ is not a valid $\mathrm{A_0PP}$ machine or it rejects 1^{n_i}. Hence, M_i^A makes a mistake for $A_i := \emptyset$.
2. f does not satisfy (3). Therefore, there exists an $\emptyset \neq A_i \subseteq \{0,1\}^n$ for which $f(A_i) \notin [0, \varepsilon T]$. Thus, M_i^A is either invalid or it wrongly accepts 1^{n_i}.

It follows that for the constructed oracle A, there exist no CM M^A and $T \in \mathrm{FP}$ satisfying (a) and (b) for L_A. □

As previously mentioned, it holds that $\mathrm{A_0PP} = \mathrm{SBQP}$ [47]. So one might ask whether we can also obtain an oracle separation for SBQP. We confirm this by showing that the inclusion $\mathrm{SBQP} \subseteq \mathrm{A_0PP}$ relativizes. We define oracle access for a quantum circuit as in [51] by allowing the

circuit to use a special oracle gate O_n^A which is defined by

$$O_n^A|x\rangle|b\rangle = |x\rangle|b \oplus A(x)\rangle \tag{4.2}$$

for all $x \in \{0,1\}^n$ and $b \in \{0,1\}$.

Lemma 4.8. *For all oracles A,* $\mathrm{SBQP}^A \subseteq \mathrm{A_0PP}^A$.

Proof. This proof is based on [47, Proposition 2.13], which in turn is based on [1, Proposition 3.2]. Let $L \in \mathrm{SBQP}^A$. Then, there exists a family of circuits $Q_n \in \mathcal{U}(\mathcal{B}^{\otimes n+q(n)})$, for some polynomial q such that for $x \in \{0,1\}^n$ and $|\psi_{\mathrm{in}}\rangle := |x\rangle \otimes |0^{q(n)}\rangle$,

$$\Pr[Q_n \text{ accepts } x] = \|\Pi_1 Q_n |\psi_{\mathrm{in}}\rangle\|_2^2 \begin{cases} \leq 2^{-p(n)-1}, & x \notin L \\ \geq 2^{-p(n)}, & x \in L \end{cases},$$

where p is a polynomial, and $\Pi_1 = |1\rangle\langle 1| \otimes I_{n+q(n)-1}$. We may assume that $Q_n = U_{t(n)} \cdots U_1$, where each U_i is either an oracle query, or a Hadamard or Toffoli gate [47]. Let $U_i' := \sqrt{2}U_i$ if U_i is a Hadamard gate, and otherwise $U_i' := U_i$. Let $Q_n' := U_{t(n)}' \cdots U_1' = \sqrt{2^k} Q_n$, where k is the number of Hadamard gates in Q_n. Then, Q_n' and U_i' only contain integer entries.

Next, we construct a function $g \in \mathrm{GapP}^A$, such that $g(x)$ equals the acceptance probability times a sufficiently large factor to make it integer. For any $z \in \{0,1\}^{n+q(n)}, i = 1, \ldots, t(n)$, we can write

$$U_i'|z\rangle = \sum_{w \in \{0,1\}^{n+q(n)}} a(i,w,z)|w\rangle.$$

Clearly, $a(i,w,z) \in \{0,1,-1\}$ and as a function $a(i,w,z) \in \mathrm{FP}^A$.

It holds that

$$2^k \|\Pi_1 Q_n |\psi_{\mathrm{in}}\rangle\|_2^2 = \|\Pi_1 Q_n' |\psi_{\mathrm{in}}\rangle\|_2^2 = \sum_{w \in \{0,1\}^{n+q(n)-1}} b_{1w}^2,$$

for

$$Q_n'|\psi_{\mathrm{in}}\rangle = \sum_{w \in \{0,1\}^{n+q(n)}} b_w |w\rangle,$$

where $b_w \in \mathbb{Z}$. We define

$$g(x) := \sum_{w \in \{0,1\}^{n+q(n)-1}} b_{1w}^2 \in \mathrm{GapP}^A,$$
$$T(x) := 2^{k-p(n)} \in \mathrm{FP}.$$

It holds that $g \in \mathrm{GapP}^A$ since the b_{1w} terms can be computed by nondeterministically choosing a path $w_0 = x0^{q(n)}, w_1, \ldots, w_{t(n)-1}, w_{t(n)} = 1w$ and multiplying the $a(i, w_i, w_{i-1})$ terms together. Hence, $L \in \mathrm{A_0PP}^A$. □

4.2.3 Query Complexity

In the following, we present an alternative oracle separation due to William Kretschmer.[2]

[2]William Kretschmer gave a proof sketch in personal communications after we came up with the presented proof for Theorem 4.7.

4.2.3.1 Approximate Degree

Let L_A be defined as in Theorem 4.7. Then, checking if $1^n \in L_A$ is equivalent to computing $\mathrm{AND}_N(X)$, where

$$\mathrm{AND}_n : \{0,1\}^n \to \{0,1\},\ x \mapsto \prod_{i=1}^{n} x_i = \begin{cases} 1, & \text{if } x = 1^n \\ 0, & \text{otherwise,} \end{cases}$$

$X \in \{0,1\}^N$, $N = 2^n$, and $X_i = A((i-1)_2)$ for $i \in [N]$, where $(m)_2$ denotes the binary encoding of $m \in \mathbb{N}$.

Definition 4.9 (Approximate Degree [52])**.** Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function, and $p : \mathbb{R}^n \to \mathbb{R}$ a polynomial (with n variables). We say p approximates f, if for every $x \in \{0,1\}^n$, we have that $|p(x) - f(x)| \leq 1/2$. The *approximate degree* of f, denoted $\widetilde{\deg}(f)$, is defined to be the minimum degree of p, over all polynomials p that approximate f.

Note that every boolean function can trivially be written as a polynomial of degree n.

Lemma 4.10 ([52])**.** $\widetilde{\deg}(\mathrm{AND}_n) = \Omega(\sqrt{n})$.

4.2.3.2 Polynomial Method

The polynomial method for quantum circuits is due to Beals et. al [8]. Let Q_X be a quantum circuit that queries the oracle string $X \in \{0,1\}^N$ at most T times. Therefore, we can write $Q_X = U_t O_t U_{t-i} O_{t-i} \cdots U_1 O_1 U_0$, where the O_i represent oracle queries of the form (4.2) and the U_i are independent of X.

Lemma 4.11 ([8])**.** *There exists a polynomial $p : \mathbb{R}^N \to \mathbb{R}$ with $\deg(p) \leq 2t$ such that*

$$\Pr[Q_X \textit{ accepts}] = p(X).$$

Now assume there exists an SBQP circuit Q_X with t oracle queries to an oracle for $X \in \{0,1\}^N$ that decides $\mathrm{AND}_N(X)$. Here, we allow the SBQP circuit to consist of an unlimited number of gates, whereas Definition 2.20 only allows $\mathrm{poly}(n)$ gates. Due to Lemma 4.11, there exists a polynomial $p : \mathbb{R}^N \to \mathbb{R}$ and polynomial $q : \mathbb{N} \to \mathbb{N}$ such that $\deg(p) \leq 2t$, and

$$\Pr[Q_X \text{ accepts}] = p(X) \begin{cases} \geq 2^{-q(n)}, & \text{if } X = 1^N \\ \leq 2^{-q(n)-1}, & \text{if } X \neq 1^N \end{cases}$$

Define $\tilde{p}(X) = p(X)/p(1^N)$. Then $\tilde{p}(1^N) = 1$ and $|\tilde{p}(X)| \leq 1/2$ for $X \neq 1^N$. Therefore, $\tilde{p}$ approximates AND_N. Thus, $t = \Omega(\sqrt{N})$ due to Lemma 4.10.

We say that AND_N has an SBQP *query complexity* $\Omega(\sqrt{N})$ as any SBQP circuit requires $\Omega(\sqrt{N})$ queries to X to decide whether $X = 1^N$.

This allows us to construct an oracle A such that $L_A \notin \mathrm{SBQP}^A$ using diagonalization as in Theorem 4.7 since a coNP machine only requires a single query to decide AND_N.

4.3 SPP vs. $\mathrm{P}^{\|\mathrm{QMA}(2)}$

In this section, we construct on oracle A such that $\mathrm{SPP}^A \not\subseteq \mathrm{P}^{\|\mathrm{QMA}(2)^A}$. Since the containments $\mathrm{SPP} \subseteq \mathrm{A_0PP}$ and $\mathrm{P}^{\mathrm{QMA[log]}} \subseteq \mathrm{P}^{\|\mathrm{QMA}(2)}$ trivially relativize, we have

$$\mathrm{SPP}^A \subseteq \mathrm{A_0PP}^A \not\subseteq \mathrm{P}^{\mathrm{QMA}^A[\log]} \subseteq \mathrm{P}^{\|\mathrm{QMA}(2)^A}.$$

Theorem 4.12. *There exists an oracle A, such that* $\mathrm{SPP}^A \not\subseteq \mathrm{P}^{\|\mathrm{QMA}(2)^A}$

Proof. For any oracle A, define $A_n := A \cap \{0,1\}^n$. Then define the language

$$L_A := \left\{1^n : |A_n| = 2^{n-1}\right\}.$$

Under the promise that $|A_n| \in \{2^{n-1}, 2^{n-1}+1\}$ for all n, it trivially holds that $L_A \in \mathrm{SPP}^A$.

Next, we will construct A such that $L_A \notin \mathrm{P}^{\|\mathrm{QMA}(2)^A}$ using diagonalization. We enumerate all Turing machines M. We may assume that M begins by writing classical descriptions of m verifiers Q_k together with the queries x_k onto a tape. The oracle then writes the answers back to the tape. For simplicity, we assume that M on input 1^n only queries strings of length n (i.e., A_n). Q_k shall have polynomially sized registers B (input), C (proof), D (ancilla), E (oracle query), and F (query answer). Hence, it holds that

$$Q_i = U_t O_n^A U_{t-1} O_n^A \cdots O_n^A U_2 O_n^A U_1,$$

where the U_i are unitary and the O_n^A are defined similarly to (4.2): For all $x \in \{0,1\}^n$ and $b \in \{0,1\}$,

$$O_n^A |\eta\rangle_{BCD} |x\rangle_E |b\rangle_F = |\eta\rangle_{BCD} |x\rangle_E |b \oplus A(x)\rangle_F$$

Let $A_n := \{0\} \times \{0,1\}^{n-1}$. In the following, we will modify A_n step by step in such a way that all oracle queries are eventually answered with 1. Let us fix some verifier Q_k, proof $|\psi\rangle$, and input $x_k \in \{0,1\}^n$. Define $|\phi_0\rangle := |x_k\rangle_B |\psi\rangle_C |0\rangle_{DEF}$ and $|\phi_i\rangle := U_i O_n \cdots O_n U_1 |\phi_0\rangle$ for $i = 1, \ldots, t$. We can write

$$|\phi_i\rangle = \sum_{y \in \{0,1\}^n} \alpha_{i,y} |y\rangle_E |\eta_{i,y}\rangle_{BCDF}.$$

Let $\varepsilon \geq 1/\operatorname{poly}(n)$. Since $\sum_y |\alpha_{i,y}|^2 = \||\phi_i\rangle\|_2^2 = 1$, there exists $z \in \{0,1\}^n \setminus A_n$, such that $|\alpha_{i,z}|^2 < \varepsilon/t$, for sufficiently large n.

Define $|\phi_i'\rangle$ analogously over $A_n' := A_n \cup \{z\}$. Trivially, $|\phi_1\rangle = |\phi_1'\rangle$. After the oracle query, we have

$$|\langle \phi_1 | O_n^\dagger O_n' | \phi_1' \rangle| = \left| \sum_{y \neq z} |\alpha_{1,y}|^2 + |\alpha_{1,z}|^2 \langle \eta_{1,z} | O_n^\dagger O_n' | \eta_{1,z}' \rangle \right| > 1 - \varepsilon/t,$$

and hence $|\langle \phi_2 | \phi_2' \rangle| > 1 - \varepsilon/t$. Continuing this argument inductively, we have $|\langle \phi_t | \phi_t' \rangle| > 1 - \varepsilon$.

For $\Pi = |1\rangle\langle 1|_{C_1} \otimes I$, we can now compare the acceptance probabilities for Q_k with oracles A_n

and A'_n for input x and proof $|\psi\rangle$:

$$\begin{aligned}
|\operatorname{Tr}(\Pi|\phi_t\rangle\langle\phi_t|) - \operatorname{Tr}(\Pi|\phi'_t\rangle\langle\phi'_t|)| &= |\operatorname{Tr}(\Pi(|\phi_t\rangle\langle\phi_t| - |\phi'_t\rangle\langle\phi'_t|))| \\
&\leq \|\Pi\|_\infty \cdot \||\phi_t\rangle\langle\phi_t| - |\phi'_t\rangle\langle\phi'_t|\|_{\mathrm{tr}} \\
&= 2\sqrt{1 - |\langle\phi_t|\phi'_t\rangle|^2} \qquad \text{(apply Lemma 2.8)} \\
&< 2\sqrt{1 - (1-\varepsilon)^2} \\
&\leq 4\sqrt{\varepsilon}
\end{aligned}$$

Thus, if the proof $|\psi\rangle$ is accepted with probability $\geq 2/3$, we can add (and by the same argument also remove) strings to A_n and either there still exists a proof accepted with probability $\geq 2/3$, or x_k becomes an invalid input as $|\psi\rangle$ is still accepted with probability $\geq 1/2$. In the latter case, we assume that the oracle still answers positively. Note that we can choose the answers for invalid oracle queries freely as we want to prove that M does not decide the language L_A.

Finally, we can describe the construction of A_n. Assume M decides language L_A with oracle access as defined above. While there exists an oracle query that is answered negatively, we alternatingly add or remove a string from A_n (at most m times). Hence, the output of M on input 1^n must change in each iteration and therefore the answer to an oracle query as well. Choosing ε such that $4\sqrt{\varepsilon} < 1/(3(m+1))$, allows us to perform at least m such operations without turning a positive query into a negative one. Note that we now must choose a z for which $|\alpha_{i,z}|^2 < \varepsilon/t$ holds in all queries, which is not an issue since there are at least $2^{n-1} - 1$ choices for z. As positive queries stay positive, at least one negative query must become positive in each iteration. We assume invalid queries are answered as negative until they become positive. Once positive, a query shall always be answered positive, as it may become invalid but not negative, by the construction of this algorithm.

Let A be the resulting oracle after at most m iterations. If the output of M is correct, we add or remove another string from A_n without changing the result of any query. Thus, M cannot decide L_A. □

Chapter 5

On $\mathrm{GSCON}_{\mathrm{exp}}$

The *ground state connectivity* problem (GSCON) introduced by Gharibian and Sikora [25] intuitively asks the following question: Given a Hamiltonian H and ground states $|\psi\rangle$ and $|\phi\rangle$, does there exist a sequence of local gates that maps $|\psi\rangle$ to $|\phi\rangle$, such that all intermediate states have low energy with respect to H? Formally, it is defined as follows.

Definition 5.1 (GSCON$(H, k, \eta_1, \eta_2, \eta_3, \eta_4, \Delta, l, m, U_\psi, U_\phi)$ [25])**.**

Input:

- A k-local Hamiltonian $H \in \mathrm{Herm}\,(\mathcal{B}^{\otimes n})$.
- $\eta_1, \eta_2, \eta_3, \eta_4, \Delta \in \mathbb{R}$ and integer $m \geq 0$, such that $\eta_2 - \eta_1 \geq \Delta$ and $\eta_4 - \eta_3 \geq \Delta$.
- Polynomial size quantum circuits U_ϕ, U_ψ generating "starting" and "target" states $|\phi\rangle$ and $|\psi\rangle$ (on input $|0^n\rangle$), respectively, satisfying $\langle\psi|H|\psi\rangle \leq \eta_1$ and $\langle\phi|H|\phi\rangle \leq \eta_1$.

Output:

YES: There exists a sequence of l-local unitaries $U_1, \ldots, U_m$ such that:

(a) (Intermediate states remain in low energy space) For all $i \in [m]$ and intermediate states $|\psi_i\rangle := U_i \cdots U_1|\psi\rangle$, it holds that $\langle\psi_i|H|\psi_i\rangle \leq \eta_1$, and

(b) (Final state is close to target state) $\||\psi_m\rangle - |\phi\rangle\|_2 \leq \eta_3$.

NO: For all l-local sequences of unitaries $U_1, \ldots, U_m$, either:

(a) (Intermediate state obtains high energy) There exists $i \in [m]$ and an intermediate state $|\psi_i\rangle$ such that $\langle\psi_i|H|\psi_i\rangle \geq \eta_2$, or

(b) (Final state far from target state) $\||\psi_m\rangle - |\phi\rangle\|_2 \geq \eta_4$.

We assume U_ψ and U_ϕ to be given as sequences of gates from a universal gate set. The numeric parameters are specified with rational entries using $O(\mathrm{poly}(n))$ bits of precision. Note that $|\psi\rangle$ and $|\phi\rangle$ are not necessarily required to be ground states.

This definition is quite flexible as it allows all parameters to be specified. For 2-local unitaries, a 5-local Hamiltonian, polynomial m and Δ, GSCON is QCMA-complete.

Theorem 5.2 ([25])**.** *There exists a polynomial p such that* GSCON *is* QCMA*-complete for $m = O(p(n))$, $\Delta = \Theta(1/m^5)$, $l = 2$, and $k \geq 5$, where n denotes the number of qubits H acts on.*

Choosing different parameters leads to PSPACE-completeness.

Theorem 5.3 ([25])**.** GSCON *is* PSPACE*-complete for $m = 2^n$, $\Delta = 2^{-(2n+4)}$, $l = 1$, $k = 3$, where n denotes the number of qubits H acts on.*

This result is a consequence of the fact that S,T-CONN is PSPACE-complete [29].

Definition 5.4 (S,T-CONN)**.** Given a 3-CNF formula ϕ and solutions $x, y \in \{0,1\}^n$ to ϕ, does there exist a sequence of strings $x_1, \ldots, x_m$, such that

- $x_1 = x$, and $x_m = y$, and
- for all $i \in [m]$, the Hamming distance between x_i and x_{i+1} is at most 1, and
- for all $i \in [m]$, x_i is a solution to ϕ?

Observe the similarity between S,T-CONN and GSCON: ϕ corresponds to H, x to $|\psi\rangle$, y to $|\phi\rangle$, and x_i to $|\psi_i\rangle$.

We now ask the question, what is the power of GSCON, for $l = 2$ and $m = 2^{\mathrm{poly}(n)}$? We call this class $\mathrm{GSCON}_{\mathrm{exp}}$.

Definition 5.5 ($\mathrm{GSCON}_{\mathrm{exp}}$)**.** $\mathrm{GSCON}_{\mathrm{exp}}$ is the union over all $GSCON(\cdots)$, where $l = 2$, $m = O(2^{p(n)})$ and $\Delta = \Omega(2^{-p(n)})$ for some polynomial p.

Let us briefly argue why we believe $\mathrm{GSCON}_{\mathrm{exp}}$ to be worthy of study. To show the containment GSCON $\subseteq$ QCMA for polynomial m and Δ, one can construct a QCMA-verifier that receives classical approximations of the unitaries $U_1, \ldots, U_m$ as proof. That technique no longer works for $\mathrm{GSCON}_{\mathrm{exp}}$. The paths through the Hamiltonian's low energy space can be of exponential length and therefore intermediate states can no longer be expressed succinctly. For that reason, we conjecture that $\mathrm{GSCON}_{\mathrm{exp}}$ may not even by contained in PSPACE.

We investigate several questions regarding $\mathrm{GSCON}_{\mathrm{exp}}$. First, we show in Section 5.1 that

$$\mathrm{PSPACE} \subseteq \mathrm{GSCON}_{\mathrm{exp}} \subseteq \mathrm{NEXP}.$$

Note that the second containment is trivial, since a NEXP-verifier can choose the unitary sequence nondeterministically. Also, the first containment is not implied by Theorem 5.3, since 2-local unitaries can flip two bits at the same time.

A natural question is whether there exists a better upper bound than NEXP. We do not know the answer to that question. One intuitive candidate would be a quantum analogue of NPSPACE, which we model as a variant of QCMA with an exponentially long proof and polynomially many qubits. However, we argue in Section 5.2 that any such construction equals NEXP.

Lastly, we show in Section 5.3, that for sufficiently large $m = 2^{\mathrm{poly}(n)}$, we can map $|\psi\rangle$ to $|\phi\rangle$ while remaining close to the Span $(|\psi\rangle, |\phi\rangle)$. We can make the distance to the span arbitrarily small by increasing m. We conclude that $\mathrm{GSCON}_{\mathrm{exp}}$ does not have any no-instances with $m = 2^{\mathrm{poly}(n)}$ and $\Delta = 2^{-o(n)}$ for sufficiently large n.

5.1 PSPACE $\subseteq \mathrm{GSCON}_{\mathrm{exp}}$

During the proof of Theorem 5.2, Gharibian and Sikora [25] show the following result.

Lemma 5.6. *Let $H \in \mathrm{Herm}\,(\mathcal{B}^{\otimes n})$ be a k'-local Hamiltonian. Consider the following promise problem Π.*

YES: *There exists a sequence $U_1, \ldots, U_m$ of l-local gates such that $U_m \cdots U_1 |0^n\rangle$ has energy at most α with respect to H.*

NO: *For all sequences $U_1, \ldots, U_m$ of l-local gates, $U_m \cdots U_1 |0^n\rangle$ has energy at least β with respect to H.*

Π *is polynomial-time reducible to* GSCON *with* $\eta_1 = \alpha$, $\eta_2 = \beta/(16m^2)$, $\eta_3 = 0$, $\eta_4 = 1/4$, $l = 2$, $k = k' + 2$, $\Delta = \eta_2 - \eta_2$, *if* $\Delta > 0$.

To prove PSPACE $\subseteq$ GSCON$_{\text{exp}}$, we combine Lemma 5.6 with two further insights. Firstly, $2^{r(n)}$ unitary 2-local gates are sufficient to construct any state $|\psi\rangle \in \mathcal{B}^n$ exactly (starting in $|0^n\rangle$) for some polynomial r [51]. Therefore, the problem Π defined in Lemma 5.6 is equivalent to the problem whether $\lambda_{\min}(H) \leq \alpha$ or $\lambda_{\min}(H) \geq \beta$ for $m = 2^{p(n)}$.

Secondly, QMA with an inverse exponential promise gap (i.e., $c - s = 2^{-\operatorname{poly}(n)}$), denoted QMA$_{\text{exp}}$, was shown by Fefferman and Lin to be PSPACE-complete [17]. They also show that k-LH with inverse exponential gap, denoted *precise* k-LH, is PSPACE-complete. Their construction leads to the following lemma, which allows us to reduce PSPACE to a precise k-LH instance with thresholds α and β, such that we can apply Lemma 5.6 to solve it in GSCON$_{\text{exp}}$ with $m = 2^{r(n)}$.

Lemma 5.7. *Any problem* Π *in* PSPACE *is reducible to a* k-LH *instance with* $\beta/\alpha \geq 2^{p(n)}$ *with* $\alpha \geq 2^{-\operatorname{poly}(n)}$, *where* $p(n)$ *is a freely chosen polynomial.*

Proof. This proof is based on [17, Theorem 2]. Π can be reduced to QMA$_{\text{exp}}$ with completeness c and soundness s, such that

$$1 - c = \varepsilon, \qquad 1 - s = -\varepsilon + 2^{-g(n)},$$

for some polynomial $g(n)$ depending on Π and any $\varepsilon = 2^{-q(n)}$ for some polynomial $q(n)$ of our choice [17].

The corresponding QMA$_{\text{exp}}$ verifier uses $T \leq h(n, \log(1/\varepsilon))$ gates, for some polynomial $h(x, y)$ [17]. Hence, QMA$_{\text{exp}}$ with completeness c and soundness s can be reduced to a 3-local Hamiltonian instance with thresholds

$$\alpha = \frac{1-c}{T+1} = \frac{\varepsilon}{T+1},$$
$$\beta = \frac{1-s}{T^3} = \frac{2^{-g(n)} - \varepsilon}{T^3}.$$

We can then choose a polynomial $q(n) \geq 2g(n)$ such that

$$\begin{aligned}
\frac{\beta}{\alpha} &= \frac{T+1}{T^3} \frac{2^{-g(n)} - \varepsilon}{\varepsilon} \\
&\geq T^{-2}\, \varepsilon^{-1}\, 2^{-g(n)-1} \\
&= \frac{2^{q(n)-g(n)-1}}{h^2(n, q(n))} \\
&\geq 2^{p(n)}.
\end{aligned}$$

□

Theorem 5.8. PSPACE $\subseteq$ GSCON$_{\text{exp}}$.

Proof. Follows directly from Lemmas 5.6 and 5.7 and the above discussion. □

5.2 A Quantum Analogue of NPSPACE

The motivation behind defining a quantum analogue of NPSPACE was to obtain a class that contains $\mathrm{GSCON}_{\mathrm{exp}}$ and is potentially strictly contained in NEXP. We model this class as a space-bounded version of QCMA, which we denote by QCMASPACE. We remark that there are potentially multiple ways to define QCMASPACE, but argue any "sensible" definition with an exponentially long classical proof equals NEXP.

The general idea for QCMASPACE is to allow the QCMA verifier to have exponentially many gates and give it access to an exponentially long proof. The exponential proof size is somewhat tricky since we do not want to have an exponential number of qubits. For this, we "stream" the proof y to the verifier, which we implement by allowing the verifier to use gates U_i, where U_i will be replaced by X if the $y_i = 1$ and I otherwise. The gates U_i must be used in ascending order of i. The circuits are then constructed by a polynomial-space Turing machine that writes the verifier's gates one after another to an auxiliary tape.

Definition 5.9 (QCMASPACE)**.** Fix polynomials $p(n)$ and $q(n)$. A promise problem Π is in QCMASPACE if there exists a polynomial-space uniform quantum circuit family $\{Q_n\}$ with the following properties:

- $Q_n \in \mathcal{U}\left(\mathcal{B}_A^{\otimes n} \otimes \mathcal{B}_C^{\otimes q(n)}\right)$. The register A is used for the input, and C contains the ancillae initialized to $|0\rangle$.
- Q_n has the form $V_t \cdots V_1$. Each V_i is either a 2-local gate from a universal set, or a *proof gate* U_j with $j \in [2^{p(n)}]$. Proof gates must occur in order. Formally, if $U_i = V_j$ and $U_k = V_l$ with $i < k$, then $j \leq l$.
- For $y \in \{0,1\}^{2^{p(n)}}$, we define Q_n^y by replacing each U_i with X if $y_i = 1$ and I otherwise.

It must hold for all $x \in \{0,1\}^n$ that

- if $x \in \Pi_{\mathrm{yes}}$, then $\exists y \in \{0,1\}^{2^{p(n)}} : \Pr[Q_n^y \text{ accepts } |x\rangle] \geq 2/3$, and
- if $x \in \Pi_{\mathrm{no}}$, then $\forall y \in \{0,1\}^{2^{p(n)}} : \Pr[Q_n^y \text{ accepts } |x\rangle] \leq 1/3$.

We observe that a PCP verifier (see Definition 2.25) can easily be simulated in QCMASPACE.

Theorem 5.10. QCMASPACE = NEXP.

Proof. The containment QCMASPACE $\subseteq$ NEXP is trivial. To show NEXP $\subseteq$ QCMASPACE, we use the fact that NEXP = PCP[poly, poly] (see Theorem 2.27). We construct a QCMASPACE verifier Q that simulates a PCP$[r, q]$ verifier. Let $T = 2^{\mathrm{poly}(n)}$ be an upper bound on the largest proof bit index accessed by M.

1. Q generates the random string $z \in \{0,1\}^{r(n)}$ by constructing a state $|+\rangle^{\otimes r(n)}$ and then measuring it in standard basis. This can be done with the usual deferred measurement technique (e.g., [51]) since $r(n)$ is polynomial (it is nontrivial to simulate an exponential number of measurements).
2. Q simulates the index computation of M and stores the indices $i_1, \ldots, i_{q(n)}$ in ancilla space.
3. For $j = 1, \ldots, T$, Q applies U_j to an ancilla $|0\rangle_c$, which maps it to $|y_j\rangle_c$. If $j = i_k$ for some k, copy y_j to a fresh ancilla. Afterwards, U_j is applied again to reset the ancilla c back to $|0\rangle_c$.
4. Simulate M with the stored proof bits to accept or reject.

Since the measured string $z \in \{0,1\}^{r(n)}$ is distributed uniformly at random, we have

$$\Pr[Q_n^y \text{ accepts } |x\rangle] = \Pr_z[M(x, y, z) = 1].$$

Note that mapping $|y_j\rangle_c$ back to $|0\rangle_c$ is no issue because the circuit is entirely classical after generating the random string. □

An alternative interpretation of Theorem 5.10 is that Savitch's theorem [56], which implies PSPACE = NPSPACE, has no quantum analogue because the space-bounded variant of BQP, denoted $\mathrm{BQ_UPSPACE}$, equals PSPACE, as shown by Fefferman and Lin [18]. $\mathrm{BQ_UPSPACE}$ is defined as BQP with polynomial-space uniformly generated quantum circuits (i.e., like QCMASPACE without a proof). Watrous [67, 65, 66] gave an earlier definition of BQPSPACE based on quantum Turing machines. The main difference between these definitions is that the quantum Turing machines may perform an exponential number of intermediate measurements, whereas that is not possible with a $\mathrm{BQ_UPSPACE}$ verifier (the subscript 'U' indicates the verifier may only perform unitary operations). The usual deferred measurement approach does not work because it requires fresh ancillae for each measurement. Both definitions nevertheless equal PSPACE. Recently, Fefferman and Remscrim [19] proved that even QMASPACE = PSPACE, where the QMASPACE verifier is an exponentially long quantum circuit that receives a polynomially-sized proof and is allowed to perform an unrestricted number of intermediate measurements. Hence, a variant of QCMASPACE with exponentially long circuit, but only polynomially sized proof, would also equal PSPACE.

5.3 Span Traversal

Suppose we are given a GSCON instance, where $l = 2$, the starting state $|\psi\rangle$ and the target state $|\phi\rangle$ are ground states of H. To determine whether we have a yes-instance, we need to check whether there exists a sequence of 2-local unitaries that maps $|\psi\rangle$ to $|\phi\rangle$ but keeps the energy of intermediate states low. Certainly, states in the span of $|\psi\rangle$ and $|\phi\rangle$ are also ground states. Therefore, we have a yes-instance if we can construct a unitary sequence, such that all intermediate states are sufficiently close to $\mathrm{Span}\,(|\psi\rangle, |\phi\rangle)$.

In this section, we show that we can indeed construct such a unitary sequence, provided we are allowed to make it exponentially long. Our general approach for this is to perform the rotation from $|\psi\rangle$ to $|\phi\rangle$ in many small rotations and then decompose each small rotation into 2-local gates (see [51]). Unfortunately, the decompositions known to us use gates like CNOT that bring intermediate states potentially far from the span.

In this section, we derive a new (approximate) decomposition of a unitary U such that individual gates are close to identity if U is close to identity (see Lemma 5.17). Afterwards, we discuss how to apply this construction to $\mathrm{GSCON_{exp}}$, even when $|\psi\rangle$ and $|\phi\rangle$ are not eigenvectors of H (see Section 5.3.4).

5.3.1 Technical Lemmas

Lemma 5.11. *For all $x \in \mathbb{R}$, it holds that $|1 - e^{ix}| \leq |x|$.*

Proof.

$$\begin{aligned}
|1 - e^{ix}| &= |1 - \cos x - i \sin x| \\
&= \sqrt{(1 - \cos x)^2 + \sin^2 x} \\
&= \sqrt{\cos^2 x - 2\cos x + 1 + \sin^2 x} \\
&= \sqrt{2 - 2\cos x} \\
&= \sqrt{2 - 2\left(1 - 2\sin^2 \frac{x}{2}\right)} && (\cos(2x) = 1 - 2\sin^2(x)) \\
&= 2\left|\sin \frac{x}{2}\right| \\
&\leq |x| && (|\sin(x)| \leq |x|)
\end{aligned}$$

□

Lemma 5.12. *Let $H \in \mathrm{Herm}(\mathbb{C}^d)$. For $U = e^{iH}$, $\|U - I\|_\infty \leq \|H\|_\infty$.*

Proof. Let $H = \sum_{j=1}^d \lambda_j |\psi_j\rangle\langle\psi_j|$ be the spectral decomposition of H. Then, the spectral decomposition of $U - I$ is given by

$$\begin{aligned}
I - e^{iH} &= \sum_{j=1}^d |\psi_j\rangle\langle\psi_j| - \sum_{j=1}^d e^{i\lambda_j} |\psi_j\rangle\langle\psi_j| \\
&= \sum_{j=1}^d \left(1 - e^{i\lambda_j}\right) |\psi_j\rangle\langle\psi_j|.
\end{aligned}$$

Therefore,

$$\|I - U\|_\infty = \max_j \left|1 - e^{i\lambda_j}\right| \leq \max_j |\lambda_j| = \|H\|_\infty,$$

where the inequality follows from Lemma 5.11. □

Lemma 5.13. *Let $U = U_m \cdots U_1$ and $V = V_m \cdots V_1$ be unitary matrices. For a submultiplicative norm $\|\cdot\|$, it holds that*

$$\|U - V\| \leq \sum_{i=1}^m \|U_i - V_i\|.$$

Proof. This proof is similar to [51, Box 4.1]. We prove the lemma by induction on m. For the base case $m = 1$ this is trivial. Now consider the case $m > 1$ and let $U' = U_{m-1} \ldots U_1, V' = V_{m-1} \ldots V_1$.

$$\begin{aligned}
\|U - V\| &= \|U_m U' - V_m V'\| \\
&= \|U_m U' - V_m U' + V_m U' - V_m V'\| && (\text{add } 0 = -V_m U' + V_m U') \\
&= \|U_m U' - V_m U'\| + \|V_m U' - V_m V'\| && (\text{triangle inequality}) \\
&= \|(U_m - V_m)U'\| + \|V_m(U' - V')\| \\
&= \|U_m - V_m\| + \|U' - V'\| && (\text{submultiplicativity}) \\
&= \|U_m - V_m\| + \sum_{i=1}^{m-1} \|U_i - V_i\| && (\text{inductive hypothesis}) \\
&= \sum_{i=1}^m \|U_i - V_i\|
\end{aligned}$$

□

Lemma 5.14 (Suzuki). *Let $H = \sum_{j=1}^{m} H_j$ be a sum of Hermitian operators such that*

$$\sum_{j=1}^{m} \|H_j\|_\infty \leq t \leq 1$$

and $n \in \mathbb{N}$. Then

$$e^{iH} = \left(\prod_{j=1}^{m} e^{iH_j/n} \right)^n + O\left(\frac{t^2}{n}\right).$$

Proof. Follows directly from [58, Theorem 3]. □

This lemma can also be used for Hamiltonian simulation, for if H is a k-local Hamiltonian, then the $e^{iH_j/n}$ terms are k-local gates. Therefore, we can simulate the evolution e^{iH} with only local gates. We can reduce the statement to the Lie-Trotter product formula

$$e^{iH_1+iH_2} = \lim_{n\to\infty} \left(e^{iH_1/n} e^{iH_2/n} \right)^n.$$

5.3.2 Decomposition of Pauli Interactions

Next, we show how to decompose operators e^{itH} for $H \in \{I, X, Y, Z\}^{\otimes n}$ into 2-local gates of the form $e^{it_j H_j}$, such that the total evolution time $\sum_j |t_j|$ is bounded by $O(t^{1/n})$. This result is due to Clinton, Bausch, and Cubitt [15].[1] [2] The main insight we use in the decomposition is as follows.

Lemma 5.15 ([15, Lemmas 6 and 8]). *Let $U = e^{itH}$ for a Hamiltonian $H = \frac{1}{2i}[h_1, h_2]$, where h_1 and h_2 anti-commute and square to identity. For $0 \leq t \leq \pi/2$, there exist $t_1, t_2 \in \mathbb{R}$ with*

$$|t_1| + |t_2| \leq \sqrt{2t},$$

and

$$U = e^{it_1 h_1} e^{it_2 h_2} e^{it_2 h_1} e^{it_1 h_2}.$$

We can also use Lemma 5.15 with negative $t \geq -\pi/2$ by applying the lemma to $-t$ and then using the inverse of the resulting decomposition ($(e^{itH})^\dagger = e^{-itH}$).

To apply this to Pauli interactions, we observe that X, Y, Z pairwise anti-commute, square to identity, and

$$[X, Y] = 2iZ, \qquad [X, Z] = 2iY, \qquad [Y, Z] = 2iX. \tag{5.1}$$

Hence, we can apply Lemma 5.15 to decompose e^{itH} for $n = 2^k + 1$ and

$$H = P_1 \otimes \cdots \otimes P_n \in \{I, X, Y, Z\}^{\otimes n}$$

into two $2^{k-1} + 1$ local evolutions as follows. Let $j = 2^{n-1} + 1$, assume $P_j = Z$, and set

$$h_1 = P_1 \otimes \cdots \otimes P_{j-1} \otimes X_j \otimes I_{j+1,\ldots,n},$$
$$h_2 = I_{1,\ldots,j-1} \otimes Y_j \otimes P_{j+1} \otimes \cdots \otimes P_n.$$

[1] We give here an alternative construction of the decomposition (still using Lemmas from [15]), since we were not able verify the correctness of the proof sketch given for [15, Equation 17].

[2] We thank Rolando Somma for pointing us to this reference.

Then,

$$\begin{aligned}[h_1, h_2] &= P_1 \otimes \cdots \otimes P_{j-1} \otimes XY_j \otimes P_{j+1} \otimes \cdots \otimes P_n \\ &\quad - P_1 \otimes \cdots \otimes P_{j-1} \otimes YX_j \otimes P_{j+1} \otimes \cdots \otimes P_n \\ &= P_1 \otimes \cdots \otimes P_{j-1} \otimes [X, Y]_j \otimes P_{j+1} \otimes \cdots \otimes P_n, \\ &= 2iH.\end{aligned}$$

The cases $P_j = X$ or $P_j = Y$ are analogous due to (5.1). The decomposition is depicted in Figure 5.1 (tensor products between the Pauli operators are omitted for conciseness).

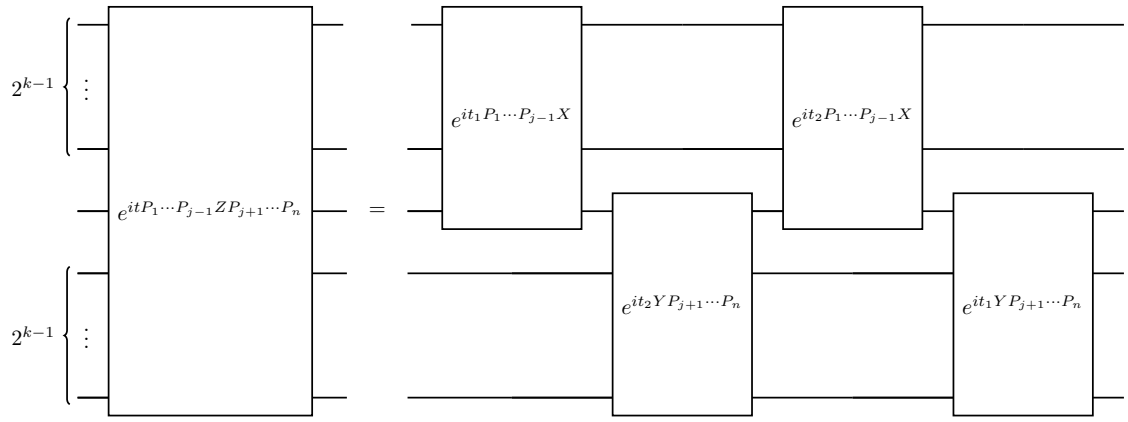


Figure 5.1: Decomposition of Pauli interactions.

For $n \neq 2^k + 1$, we cannot always choose the split j to be exactly in the middle, but the resulting interactions will still be at most $(2^{k-1} + 1)$-local, provided $n \leq 2^k + 1$. Applying this decomposition recursively we show:

Lemma 5.16. *Let $H \in \{I, X, Y, Z\}^{\otimes n}$ with $n \in (2^{k-1} + 1,\, 2^k + 1]$, and $t \in \mathbb{R}$ with $8|t|^{2^{-k}} \leq \pi/2$. There exists a decomposition of $e^{itH} = \Pi_{j=1}^m e^{it_j H_j}$, where the H_j are 2-local Pauli matrices, $m \leq 4^k = O(n^2)$, and $\sum_{i=1}^m |t_i| = O(n^2 |t|^{2^{-k}}) = O(|t|^{1/2n})$.*

Proof. We construct the 2-local $H_1, \ldots, H_m$ by applying Lemma 5.15 recursively as outlined above. $e^{itH} = \Pi_{j=1}^m e^{it_j H_j}$ follows from the correctness of Lemma 5.15. After each recursion, we have interactions that are at most $(2^{k-1} + 1)$-local. Hence, after k recursions, only 2-local interactions remain and we are done. Since each recursion increases the number of interactions by a factor of 4, we have $m \leq 4^k = O(n^2)$.

By Lemma 5.15, a recursion constructs new interactions, each with a pulse time of at most

$$|t_1| + |t_2| \leq \sqrt{2t}.$$

This gives us a recurrence for an upper bound on the individual pulse times after r recursions:

$$T(r) = \begin{cases} |t|, & \text{if } r = 0 \\ \sqrt{2T(r-1)}, & \text{if } r > 0 \end{cases}.$$

Using induction on r, we show that

$$T(r) = 2^{1-2^{-r}}|t|^{2^{-r}}.$$

For $r = 0$, we have $T(0) = |t|$. For $r > 0$, we have

$$\begin{aligned} T(r) &= \sqrt{2T(r-1)} \\ &= \left(2 \cdot 2^{1-2^{-r+1}}|t|^{2^{-r+1}}\right)^{1/2} \\ &= 2^{1/2} \cdot 2^{1/2-2^{-r}}|t|^{2^{-r}} \\ &= 2^{1-2^{-r}}|t|^{2^{-r}}. \end{aligned}$$

Hence, the individual pulse times after k recursions are bounded by $T(k) \leq 8|t|^{2^{-k}}$. The total pulse time is then bounded by $mT(k) = O(n^2|t|^{1/2n})$. □

Remark. Our decomposition only uses polynomially many gates, whereas it appears to us that the construction sketched in [15] uses exponentially many. This might be of interest for physical applications. We also only require their Lemmas 6 and 8, without having to use the more complex Lemmas 7 and 9. Their decomposition has a total pulse time of $O(|t|^{1/n})$.

5.3.3 General Decomposition

Next, we show how to use Lemma 5.16 to decompose general unitaries.

Lemma 5.17. *Let $U = e^{iH}$ for Hermitian $H \in \mathrm{Herm}(\mathbb{C}^d)$, $d = 2^n$ with*

$$\|H\|_\infty =: \varepsilon < c_n := (\pi/16)^{2n}.$$

There exists an approximate decomposition

$$U = U_m \cdots U_1 + O\left(d^3\varepsilon^2\right)$$

into $m \leq 2^{O(n)}$ 2-local unitaries, such that

$$\sum_{j=1}^{m} \|I - U_j\|_\infty = O\left(n^2 d^2 \varepsilon^{1/2n}\right) \tag{5.2}$$

Proof. We write H in the Pauli basis:

$$H = \sum_{j=1}^{d^2} \alpha_j P_j,$$

where $\alpha_j \in \mathbb{R}$ and $P_j \in \{I, X, Y, Z\}^{\otimes n}$ for all $j \in [d^2]$. By viewing H as the sum of orthogonal vectors with $\|P_j\|_{\mathrm{F}} = \sqrt{d}$, we get

$$\|H\|_{\mathrm{F}} = \sqrt{d \sum_{j=1}^{d^2} \alpha_j^2} \leq \sqrt{d}\|H\|_\infty, \tag{5.3}$$

where the inequality follows from Lemma 2.7. Therefore, $|\alpha_j| \leq \varepsilon$ for all $j \in [d^2]$. It holds that

$$\begin{aligned}
\sum_{j=1}^{d^2} \|\alpha_j P_j\|_\infty &= \sum_{j=1}^{d^2} |\alpha_j| \\
&\leq d\sqrt{\sum_{j=1}^{d^2} \alpha_j^2} && \text{(apply Lemma 2.5)} \\
&\leq d\|H\|_\infty && \text{(apply (5.3))} \\
&\leq 1.
\end{aligned}$$

By Lemma 5.14,

$$U = e^{iH} = \prod_{j=1}^{d^2} e^{i\alpha_j P_j} + O\left(d^3\varepsilon^2\right).$$

Lemma 5.16 allows us to decompose each term $e^{i\alpha_j P_j}$ into $m_j = O(n^2)$ 2-local unitaries

$$e^{i\alpha_j P_j} = \prod_{k=1}^{m_j} e^{it_{j,k}H_{j,k}}$$

with an evolution time of $\sum_{k=1}^{m_j} |t_{j,k}| = O(n^2|\alpha_j|^{1/2n})$. We get the complete decomposition

$$U = e^{iH} = \prod_{j=1}^{d^2} \prod_{k=1}^{m_j} e^{it_{j,k}H_{j,k}} + O(d^3\varepsilon^2),$$

with a total evolution time of

$$O\left(\sum_{j=1}^{d^2} |\alpha_j|^{1/2n}\right) = O\left(n^2 d^2 \varepsilon^{1/2n}\right).$$

(5.2) follows from Lemma 5.12 as

$$\|I - e^{it_{j,k}H_{j,k}}\|_\infty \leq \|t_{j,k}H_{j,k}\|_\infty = t_{j,k}.$$

□

We remark, that we usually choose $\varepsilon \ll c_n$ in order to make (5.2) small. Furthermore, the above decomposition is approximate. It appears to be an open question whether a similar result is achievable with an exact decomposition.

We can use this decomposition to map between two close vectors $|\psi\rangle$ and $|\phi\rangle$ while bounding the distance of intermediate states from $|\psi\rangle$. Although we do not use this construction, it may still be of independent interest. Since the error scales as $O(\varepsilon^2)$, we can use it to map a quantum state along any continuous path with an arbitrarily small error (at the cost of more gates).

Lemma 5.18. *Let $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$ with $d = 2^n$. Let $\||\psi\rangle - |\phi\rangle\|_2 \leq \varepsilon < c_n$. There exists a sequence of 2-local unitaries $U = U_m \cdots U_1$ with $m \leq 2^{\mathrm{poly}(n)}$, such that*

(1) $\||\phi\rangle - U|\psi\rangle\|_2 = O(d^3\varepsilon^2)$, and

(2) for all $i \in [m]$, $\||\psi\rangle - U_i \cdots U_1|\psi\rangle\|_2 = O(n^2 d^2 \varepsilon^{1/2n})$.

Proof. Let $\theta = \cos^{-1} \operatorname{Re}(\langle\psi|\phi\rangle)$ be the angle between $|\psi\rangle$ and $|\phi\rangle$. After a suitable change of basis W, we have

$$\begin{aligned} W|\psi\rangle &= |0\rangle, \\ W|\phi\rangle &= \cos(\theta)|0\rangle + \sin(\theta)|1\rangle. \end{aligned}$$

Hence, we only need to apply the rotation matrix $R(\theta)$ (extended to d dimensions as in (5.4)) to map from $W|\psi\rangle$ to $W|\phi\rangle$. Let $V = W^\dagger R(\theta) W$. V has the same eigenvalues as $R(\theta)$, namely $e^{i\theta}, e^{-i\theta}, 1$. Hence, $V = e^{iH}$ for $\|H\|_\infty = |\theta|$.

To apply Lemma 5.17, we need to bound θ. We have for $|\theta| < 1$,

$$\begin{aligned} \||\psi\rangle - |\phi\rangle\|_2 &= \sqrt{2 - 2\operatorname{Re}(\langle\phi|\psi\rangle)} && \text{(apply Lemma 2.6)} \\ &= \sqrt{2 - 2\cos\theta} \\ &\geq \sqrt{\theta^2 - \theta^4/12} && \text{(Taylor expansion)} \\ &\geq |\theta|/2. \end{aligned}$$

Hence, $|\theta| \leq 2\varepsilon$. Let $U = U_m \cdots U_1$ from Lemma 5.17. (1) and (2) follow from Lemma 5.13. □

5.3.4 Application to $\mathrm{GSCON}_{\exp}$

We now use Lemma 5.17 to construct a sequence of unitaries to map from the starting state $|\psi\rangle$ to the target state $|\phi\rangle$ while remaining in low energy space.

Theorem 5.19. *Let $H \in \operatorname{Herm}\left(\mathbb{C}^d\right)$, $d = 2^n$ with $0 \preccurlyeq H \preccurlyeq I$, $|\psi\rangle, |\phi\rangle \in \mathbb{C}^d$ with $\langle\psi|H|\psi\rangle \leq \eta$ and $\langle\phi|H|\phi\rangle \leq \eta$. For $\Delta \geq 2^{-\operatorname{poly}(n)}$, there exists a sequence of 2-local unitary gates $U = U_m \cdots U_1$ with $m \leq 2^{\operatorname{poly}(n)}$ such that*

(1) $\|U|\psi\rangle - |\phi\rangle\|_2 \leq \Delta$, and

(2) for all $i \in [m]$, $\langle\psi_i|H|\psi_i\rangle \leq \eta + \Delta$, where $|\psi_i\rangle := U_i \cdots U_1|\psi\rangle$.

Proof. The idea is to first rotate $|\psi\rangle$ onto a ground state $|\mu\rangle$ and then use the same method to rotate $|\mu\rangle$ to $|\phi\rangle$. This will leave all intermediate states at an energy below η (in practice it might exceed η since the construction is approximate). Let $|\lambda_1\rangle, \ldots, |\lambda_d\rangle$ be an orthonormal eigenbasis of H. We assume $|\mu\rangle = |\lambda_1\rangle$ is a ground state. We can write

$$|\psi\rangle = \cos(\theta)|\mu\rangle + \sin(\theta)|\nu\rangle,$$

where $|\nu\rangle \in \operatorname{Span}\left(|\lambda_2\rangle, \ldots, |\lambda_d\rangle\right)$.

We argue that increasing the amplitude on $|\mu\rangle$ and decreasing the amplitude on $|\nu\rangle$ cannot increase the energy. Let $|\psi'\rangle = \cos(\theta')|\mu\rangle + \sin(\theta')|\nu\rangle$ with $|\cos\theta'| > |\cos\theta|$. Then, for $\lambda = \lambda_{\min}(H)$,

$$\begin{aligned} \langle\psi|H|\psi\rangle &= \cos^2(\theta)\lambda + \sin^2(\theta)\langle\nu|H|\nu\rangle \\ &= \cos^2(\theta)\lambda + (1 - \cos^2(\theta))\langle\nu|H|\nu\rangle \\ &< \cos^2(\theta')\lambda + (1 - \cos^2(\theta'))\langle\nu|H|\nu\rangle && (\lambda \leq \langle\nu|H|\nu\rangle) \\ &= \langle\psi'|H|\psi'\rangle. \end{aligned}$$

To rotate $|\psi\rangle$ to $|\mu\rangle$, we can apply a rotation matrix $V = W^\dagger R(\alpha) W$ t times, where W is a suitable

change of basis matrix, and $R(\alpha)$ is the rotation matrix extended to d dimensions with $\alpha := -\theta/t$:

$$R(\alpha) = \left(\begin{array}{cc|c} \cos\alpha & -\sin\alpha & \\ \sin\alpha & \cos\alpha & 0 \\ \hline & 0 & I \end{array}\right) \tag{5.4}$$

Therefore, V has the eigenvalues $e^{i\alpha}, e^{-i\alpha}, 1$ and we have $V = e^{iH}$ with $\|H\|_\infty \leq |\alpha| = \varepsilon = O(1/t)$. It further holds that $\langle\psi_j|H|\psi_j\rangle \leq \eta$ for $j \in [t]$, $|\psi_j\rangle := V^j|\psi\rangle$, and $|\mu\rangle = |\psi_t\rangle$.

Choosing a sufficiently large $t = \mathrm{poly}(d)$, Lemma 5.17 gives us a decomposition $V = U_m \cdots U_1 + O(d^3\varepsilon^2)$ with

$$\sum_{j=1}^{m} \|I - U_j\|_\infty = O\left(n^2 d^2 \varepsilon^{1/2n}\right) =: \gamma \tag{5.5}$$

for $m \leq 2^{\mathrm{poly}(n)}$.

Let $V' := U_m \cdots U_1$ and $|\psi'_j\rangle := V'^j|\psi\rangle$. Due to Lemma 5.13, for all $j \in [t]$,

$$\||\psi_j\rangle - |\psi'_j\rangle\|_2 = O(jd^3\varepsilon^2) = O(d^3\varepsilon) =: \delta.$$

Then since $\|H\|_\infty \leq 1$,

$$\begin{aligned} \langle\psi'_j|H|\psi'_j\rangle &= \langle\psi'_j|H|\psi_j\rangle + \langle\psi'_j|H\left(|\psi'_j\rangle - |\psi_j\rangle\right) \\ &= \langle\psi_j|H|\psi_j\rangle + \left(\langle\psi'_j| - \langle\psi_j|\right)H|\psi_j\rangle + \langle\psi'_j|H\left(|\psi'_j\rangle - |\psi_j\rangle\right) \\ &\leq \eta + 2\delta \end{aligned}$$

For $|\psi'_{j,k}\rangle := U_k \cdots U_1|\psi'_j\rangle$, we have due to Lemma 5.13 and (5.5)

$$\||\psi'_{j,k}\rangle - |\psi'_j\rangle\|_2 \leq \gamma,$$

and by the triangle inequality

$$\||\psi'_{j,k}\rangle - |\psi_j\rangle\|_2 \leq \gamma + \delta.$$

Choosing t such that $\gamma + 2\delta \leq \Delta$, we satisfy (1) and (2) while mapping $|\psi\rangle$ to $|\mu\rangle$.

We complete the proof by applying the inverse of the above construction, but for $|\phi\rangle$ to $|\mu\rangle$. This increases the error terms by a factor of 2. Hence, we choose t such that $2\gamma + 4\delta \leq \Delta$. □

Note that the theorem easily generalizes to any $0 \preccurlyeq H \preccurlyeq 2^{\mathrm{poly}(n)}I$. We further remark that our small pulse time decomposition really is crucial for this result since $R(\alpha)$ and W are not 2-local in general. Thus, the usual decompositions would introduce gates with large pulse times that could bring intermediate states far from the span.

5.3.4.1 Relation to the 1-Local Case

We have seen that any $\mathrm{GSCON}_{\mathrm{exp}}$ instance becomes either a yes-instance or an invalid instance if we make m sufficiently large. It is not obvious to us whether something similar is possible for $l = 1$ (i.e., only 1-local unitaries can be applied). Of course, we cannot decompose arbitrary rotations into 1-local unitaries. This raises the question whether the $l = 1$ case is PSPACE-hard for unbounded m. Also, is it still contained in PSPACE?

This apparent difference between $l = 1$ and $l = 2$ really seems to be due to the continuous state space in the quantum setting. Allowing k bitflips in each step of S, T-CONN (i.e., the Hamming distance in each step is allowed to be k), does not change the power of S, T-CONN: Given boolean formula ϕ, we can construct ϕ' by adding $k-1$ additional copies of each variable and constraints to verify that all k copies have the same value. Then, going from one satisfying assignment to another requires at least k bitflips, because all copies of a variable need to be changed.

5.3.4.2 Relation to the Traversal Lemma

Gharibian and Sikora [25] use the *Traversal Lemma* as an important tool to show that GSCON is QCMA-complete. Two states $|u\rangle, |w\rangle \in \mathcal{B}^{\otimes n}$ are said to be *k-orthogonal* if for all k-local unitaries U, we have $\langle w|U|v\rangle = 0$. Two subspaces $S, T \subseteq \mathcal{B}^{\otimes n}$ are called k-orthogonal if any pair of vectors $|v\rangle \in S, |w\rangle \in T$ is k-orthogonal.

Lemma 5.20 (Traversal Lemma [25])**.** *Let $S, T \subseteq \mathcal{B}^{\otimes n}$ be k-orthogonal subspaces. Let $|v\rangle \in S, |w\rangle \in T$ and consider a sequence of unitaries $U_1, \ldots, U_m$ with*

$$\| |w\rangle - U_m \cdots U_1 |v\rangle \|_2 \leq \varepsilon < 1/2.$$

Let $|v_i\rangle := U_i \cdots U_1 |v\rangle$ and $P := I - \Pi_S - \Pi_T$. Then, there exists an $i \in [m]$ such that

$$\langle v_i|P|v_i\rangle \geq \left(\frac{1-2\varepsilon}{2m}\right)^2.$$

Gharibian and Sikora provide an example for which the Traversal Lemma is tight by explicitly constructing a gate sequence to map $|000\rangle$ to $|111\rangle$ with $\langle v_i|P|v_i\rangle \leq \Delta$ for $m = O(1/\Delta^2)$.

Our Theorem 5.19 constructs such a sequence in general, although it is not as tight, because m is only polynomial in Δ^{-1} and exponential in n.

Bibliography

[1] S. Aaronson. "Quantum computing, postselection, and probabilistic polynomial-time." In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 461.2063 (2005), pp. 3473–3482. DOI: 10.1098/rspa.2005.1546.

[2] D. Aharonov, M. Ben-Or, F. G. S. L. Brandao, and O. Sattath. "The Pursuit For Uniqueness: Extending Valiant-Vazirani Theorem to the Probabilistic and Quantum Settings." In: *arXiv:0810.4840 [quant-ph]* (Oct. 2008). arXiv: 0810.4840.

[3] D. Aharonov, A. B. Grilo, and Y. Liu. "StoqMA vs. MA: the power of error reduction." In: *arXiv:2010.02835 [quant-ph]* (Oct. 2020). arXiv: 2010.02835.

[4] A. Ambainis. "On Physical Problems that are Slightly More Difficult than QMA." In: *2014 IEEE 29th Conference on Computational Complexity (CCC)*. 2014, pp. 32–43.

[5] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. "Proof verification and the hardness of approximation problems." In: *Journal of the ACM* 45.3 (May 1998), pp. 501–555. ISSN: 0004-5411. DOI: 10.1145/278298.278306.

[6] S. Arora and S. Safra. "Probabilistic checking of proofs: a new characterization of NP." In: *Journal of the ACM* 45.1 (Jan. 1998), pp. 70–122. ISSN: 0004-5411. DOI: 10.1145/273865.273901.

[7] L. Babai, L. Fortnow, and C. Lund. "Nondeterministic exponential time has two-prover interactive protocols." In: *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*. Oct. 1990, 16–25 vol.1. DOI: 10.1109/FSCS.1990.89520.

[8] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. "Quantum lower bounds by polynomials." In: *Journal of the ACM* 48.4 (July 2001), pp. 778–797. ISSN: 0004-5411. DOI: 10.1145/502090.502097.

[9] R. Beigel, L. Hemachandra, and G. Wechsung. "On the power of probabilistic polynomial time: $P^{NP[log]} \subseteq PP$." In: *[1989] Proceedings. Structure in Complexity Theory Fourth Annual Conference*. June 1989, pp. 225–227. DOI: 10.1109/SCT.1989.41828.

[10] H. L. Bodlaender. "A linear time algorithm for finding tree-decompositions of small treewidth." In: *Proceedings of the twenty-fifth annual ACM symposium on Theory of Computing*. STOC '93. San Diego, California, USA: Association for Computing Machinery, June 1993, pp. 226–234. ISBN: 9780897915915. DOI: 10.1145/167088.167161.

[11] A. D. Bookatz. "QMA-complete problems." In: *Quantum Information & Computation* 14.5&6 (Apr. 2014), pp. 361–383. ISSN: 1533-7146.

[12] S. Bravyi, A. J. Bessen, and B. M. Terhal. "Merlin-Arthur Games and Stoquastic Complexity." In: (Dec. 2006). arXiv: quant-ph/0611021 [quant-ph].

[13] S. R. Buss and L. Hay. "On truth-table reducibility to SAT." In: *Information and Computation* 91.1 (1991), pp. 86–102. ISSN: 0890-5401. DOI: 10.1016/0890-5401(91)90075-D.

[14] A. Chailloux and O. Sattath. "The Complexity of the Separable Hamiltonian Problem." In: *2012 IEEE 27th Conference on Computational Complexity*. ISSN: 1093-0159. June 2012, pp. 32–41. DOI: 10.1109/CCC.2012.42.

[15] L. Clinton, J. Bausch, and T. Cubitt. "Hamiltonian Simulation Algorithms for Near-Term Quantum Hardware." In: *arXiv:2003.06886 [quant-ph]* (Mar. 2020). arXiv: 2003.06886.

[16] S. A. Cook. "The complexity of theorem-proving procedures." In: *Proceedings of the third annual ACM symposium on Theory of computing*. STOC '71. Shaker Heights, Ohio, USA: Association for Computing Machinery, May 1971, pp. 151–158. ISBN: 9781450374644. DOI: 10.1145/800157.805047.

[17] B. Fefferman and C. Lin. "Quantum Merlin Arthur with Exponentially Small Gap." In: (Jan. 2016). arXiv: 1601.01975 [quant-ph].

[18] B. Fefferman and C. Y.-Y. Lin. "A Complete Characterization of Unitary Quantum Space." In: *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Ed. by A. R. Karlin. Vol. 94. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018, 4:1–4:21. ISBN: 9783959770606. DOI: 10.4230/LIPIcs.ITCS.2018.4. arXiv: 1909.05981 [quant-ph].

[19] B. Fefferman and Z. Remscrim. "Eliminating Intermediate Measurements in Space-Bounded Quantum Computation." In: *arXiv:2006.03530 [quant-ph]* (June 2020). arXiv: 2006.03530.

[20] S. A. Fenner, L. J. Fortnow, and S. A. Kurtz. "Gap-definable counting classes." In: *Journal of Computer and System Sciences* 48.1 (Feb. 1994), pp. 116–148. ISSN: 0022-0000. DOI: 10.1016/S0022-0000(05)80024-8.

[21] L. Fortnow and N. Reingold. "PP Is Closed under Truth-Table Reductions." In: *Information and Computation* 124.1 (Jan. 1996), pp. 1–6. ISSN: 0890-5401. DOI: 10.1006/inco.1996.0001.

[22] S. Gharibian. *Lecture notes: Quantum Complexity Theory*. Paderborn University, 2019.

[23] S. Gharibian, Y. Huang, Z. Landau, and S. W. Shin. "Quantum Hamiltonian Complexity." In: *Foundations and Trends® in Theoretical Computer Science* 10.3 (Oct. 2015), pp. 159–282. ISSN: 1551-305X. DOI: 10.1561/0400000066.

[24] S. Gharibian, S. Piddock, and J. Yirka. "Oracle Complexity Classes and Local Measurements on Physical Hamiltonians." In: *37th International Symposium on Theoretical Aspects of Computer Science (STACS 2020)*. Ed. by C. Paul and M. Bläser. Vol. 154. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 20:1–20:37. ISBN: 978-3-95977-140-5. DOI: 10.4230/LIPIcs.STACS.2020.20. arXiv: 1909.05981 [quant-ph].

[25] S. Gharibian and J. Sikora. "Ground State Connectivity of Local Hamiltonians." In: *ACM Transactions on Computation Theory* 10.2 (Apr. 2018), 8:1–8:28. ISSN: 1942-3454. DOI: 10.1145/3186587. arXiv: 1409.3182 [quant-ph].

[26] S. Gharibian and J. Yirka. "The complexity of simulating local measurements on quantum systems." In: *Quantum* 3 (Sept. 2019), p. 189. DOI: 10.22331/q-2019-09-30-189.

[27] J. T. Gill. "Computational complexity of probabilistic Turing machines." In: *Proceedings of the sixth annual ACM symposium on Theory of computing*. STOC '74. Seattle, Washington, USA: Association for Computing Machinery, Apr. 1974, pp. 91–95. ISBN: 9781450374231. DOI: 10.1145/800119.803889.

[28] G. Golub. *Matrix computations*. Baltimore: Johns Hopkins University Press, 1996. ISBN: 080185413X.

[29] P. Gopalan, P. G. Kolaitis, E. N. Maneva, and C. H. Papadimitriou. "The Connectivity of Boolean Satisfiability: Computational and Structural Dichotomies." In: *Automata, Languages and Programming*. Ed. by M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2006, pp. 346–357. ISBN: 9783540359050. DOI: 10.1007/11786986_31.

[30] D. Gosset, J. C. Mehta, and T. Vidick. "QCMA hardness of ground space connectivity for commuting Hamiltonians." In: *Quantum* 1 (July 2017), p. 16. DOI: 10.22331/q-2017-07-14-16.

[31] G. Gottlob. "NP trees and Carnap's modal logic." In: *Journal of the ACM* 42.2 (Mar. 1995), pp. 421–457. ISSN: 0004-5411. DOI: 10.1145/201019.201031.

[32] H. Gruber. "On Balanced Separators, Treewidth, and Cycle Rank." In: *arXiv:1012.1344 [cs, math]* (Jan. 2013). arXiv: 1012.1344.

[33] A. W. Harrow and A. Montanaro. "Testing product states, quantum Merlin-Arthur games and tensor optimisation." In: *Journal of the ACM* 60.1 (Feb. 2013), pp. 1–43. ISSN: 0004-5411, 1557-735X. DOI: 10.1145/2432622.2432625. arXiv: 1001.0017.

[34] J. Hartmanis. "Sparse Complete Sets for NP and the Optimal Collapse of the Polynomial Hierarchy." In: *Current Trends in Theoretical Computer Science*. WORLD SCIENTIFIC, June 1993, pp. 403–411. DOI: 10.1142/9789812794499_0029.

[35] L. A. Hemachandra. "The strong exponential hierarchy collapses." In: *Journal of Computer and System Sciences* 39.3 (Dec. 1989), pp. 299–322. ISSN: 0022-0000. DOI: 10.1016/0022-0000(89)90025-1.

[36] E. Hemaspaandra, L. A. Hemaspaandra, and J. Rothe. "Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP." In: *Automata, Languages and Programming*. Ed. by P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1997, pp. 214–224. ISBN: 9783540691945. DOI: 10.1007/3-540-63165-8_179.

[37] W. Hoeffding. "Probability Inequalities for Sums of Bounded Random Variables." In: *Journal of the American Statistical Association* 58.301 (Mar. 1963), pp. 13–30. ISSN: 0162-1459. DOI: 10.1080/01621459.1963.10500830.

[38] R. M. Karp. "Reducibility among Combinatorial Problems." In: ed. by R. E. Miller, J. W. Thatcher, and J. D. Bohlinger. The IBM Research Symposia Series. Boston, MA: Springer US, 1972, pp. 85–103. ISBN: 9781468420012. DOI: 10.1007/978-1-4684-2001-2_9.

[39] J. Kempe, A. Kitaev, and O. Regev. "The Complexity of the Local Hamiltonian Problem." In: *FSTTCS 2004: Foundations of Software Technology and Theoretical Computer Science*. Ed. by K. Lodaya and M. Mahajan. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2005, pp. 372–383. ISBN: 9783540305385. DOI: 10.1007/978-3-540-30538-5_31.

[40] J. Kempe and O. Regev. "3-local Hamitonian is QMA-complete." In: *Quantum Information & Computation* 3.3 (May 2003), pp. 258–264. ISSN: 1533-7146.

[41] A. Y. Kitaev. "Unpaired Majorana fermions in quantum wires." In: *Physics-Uspekhi* 44.10S (Oct. 2001), pp. 131–136. ISSN: 1063-7869. DOI: 10.1070/1063-7869/44/10S/S29.

[42] A. Y. Kitaev. "Fault-tolerant quantum computation by anyons." In: *Annals of Physics* 303.1 (Jan. 2003), pp. 2–30. ISSN: 0003-4916. DOI: 10.1016/S0003-4916(02)00018-0.

[43] A. Y. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. USA: American Mathematical Society, 2002. ISBN: 0821832298.

[44] A. Y. Kitaev and C. Laumann. "Topological phases and quantum computation." In: *arXiv:0904.2771 [cond-mat, physics:quant-ph]* (Apr. 2009). arXiv: 0904.2771.

[45] A. Y. Kitaev and J. Watrous. "Parallelization, amplification, and exponential time simulation of quantum interactive proof systems." In: *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. STOC '00. Portland, Oregon, USA: Association for Computing Machinery, May 2000, pp. 608–617. ISBN: 9781581131840. DOI: 10.1145/335305.335387.

[46] M. W. Krentel. "Generalizations of Opt P to the polynomial hierarchy." In: *Theoretical Computer Science* 97.2 (Apr. 1992), pp. 183–198. ISSN: 0304-3975. DOI: 10.1016/0304-3975(92)90073-O.

[47] G. Kuperberg. "How Hard Is It to Approximate the Jones Polynomial?" In: *Theory of Computing* 11 (June 2015), pp. 183–219. DOI: 10.4086/toc.2015.v011a006.

[48] L. A. Levin. "Universal sequential search problems." In: *Problems of Information Transmission* 9.3 (1973), pp. 265–266.

[49] A. A. Markov. "On a question by D. I. Mendeleev." In: *Zap. Imp. Akad. Nauk. St. Petersburg* 62 (1890), pp. 1–24.

[50] C. Marriott and J. Watrous. "Quantum Arthur–Merlin games." In: *computational complexity* 14.2 (June 2005), pp. 122–152. ISSN: 1420-8954. DOI: 10.1007/s00037-005-0194-x.

[51] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 10th. USA: Cambridge University Press, 2011. ISBN: 1107002176.

[52] N. Nisan and M. Szegedy. "On the degree of boolean functions as real polynomials." In: *computational complexity* 4.4 (Dec. 1994), pp. 301–313. ISSN: 1420-8954. DOI: 10.1007/BF01263419.

[53] T. J. Osborne. "Hamiltonian complexity." In: *Reports on Progress in Physics* 75.2 (Jan. 2012), p. 022001. ISSN: 0034-4885. DOI: 10.1088/0034-4885/75/2/022001.

[54] C. H. Papadimitriou and S. K. Zachos. "Two remarks on the power of counting." In: *Theoretical Computer Science*. Ed. by A. B. Cremers and H.-P. Kriegel. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1982, pp. 269–275. ISBN: 9783540394211. DOI: 10.1007/BFb0036487.

[55] E. A. Rakhmanov. "Bounds for Polynomials with a Unit Discrete Norm." In: *Annals of Mathematics* 165.1 (2007), pp. 55–88. ISSN: 0003486X.

[56] W. J. Savitch. "Relationships between nondeterministic and deterministic tape complexities." In: *Journal of Computer and System Sciences* 4.2 (Apr. 1970), pp. 177–192. ISSN: 0022-0000. DOI: 10.1016/S0022-0000(70)80006-X.

[57] L. J. Stockmeyer. "The polynomial-time hierarchy." In: *Theoretical Computer Science* 3.1 (Oct. 1976), pp. 1–22. ISSN: 0304-3975. DOI: 10.1016/0304-3975(76)90061-X.

[58] M. Suzuki. "Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems." In: *Communications in Mathematical Physics* 51.2 (1976), pp. 183–190. ISSN: 0010-3616, 1432-0916.

[59] S. Toda. "PP is as hard as the polynomial-time hierarchy." In: *SIAM Journal on Computing* 20.5 (Oct. 1991), pp. 865–877. ISSN: 0097-5397. DOI: 10.1137/0220053.

[60] L. G. Valiant. "The complexity of computing the permanent." In: *Theoretical Computer Science* 8.2 (Jan. 1979), pp. 189–201. ISSN: 0304-3975. DOI: 10.1016/0304-3975(79)90044-6.

[61] M. Vyalyi. "QMA=PP implies that PP contains PH." In: *Electronic Colloquium on Computational Complexity (ECCC)* 10 (May 2003).

[62] K. Wagner. "Bounded query computations." In: *[1988] Proceedings. Structure in Complexity Theory Third Annual Conference* (1988). DOI: 10.1109/SCT.1988.5286.

[63] K. W. Wagner. "More complicated questions about maxima and minima, and some closures of NP." In: *Theoretical Computer Science* 51.1 (Jan. 1987), pp. 53–80. ISSN: 0304-3975. DOI: 10.1016/0304-3975(87)90049-1.

[64] K. W. Wagner. "The complexity of combinatorial problems with succinct input representation." In: *Acta Informatica* 23.3 (June 1986), pp. 325–356. ISSN: 1432-0525. DOI: 10.1007/BF00289117.

[65] J. Watrous. "On the complexity of simulating space-bounded quantum computations." In: *computational complexity* 12.1 (June 2003), pp. 48–84. ISSN: 1420-8954. DOI: 10.1007/s00037-003-0177-8.

[66] J. Watrous. "Quantum Computational Complexity." In: *arXiv:0804.3401 [quant-ph]* (Apr. 2008). arXiv: 0804.3401.

[67] J. Watrous. "Space-Bounded Quantum Complexity." In: *Journal of Computer and System Sciences* 59.2 (Oct. 1999), pp. 281–326. ISSN: 0022-0000. DOI: 10.1006/jcss.1999.1655.