Quantum Hamiltonian Complexity
Praveen Balaji
Advanced Quantum Mechanics Winter 2021

Contents

1 Introduction and Motivation 1

2 Quantum Hamiltonian Complexity 2
   2.1 Classical Complexity Theory ............................. 2
   2.2 Quantum Complexity Theory .............................. 4
   2.3 Cornerstone of Quantum Hamiltonian Complexity - k-local Hamiltonian 5
      2.3.1 Definition and Complexity Results .................. 5
      2.3.2 Quantum PCP Conjecture ............................. 9
   2.4 Developments of Interest ................................ 10

1 Introduction and Motivation

One of the historic and most fundamental goals of physics is to extract and understand the properties of quantum materials. With the immense number of degrees of freedom within each material, often times directly solving for the Schrödinger equation to get a closed form for the Hamiltonian is simply impossible. So, instead, we typically resort to the grade-school scientific method and take the following steps

1) Formulate a theoretical Hamiltonian
2) Make a prediction about some property of the system based on the theoretical Hamiltonian
3) Test for that property and modify formulated Hamiltonian accordingly

Unfortunately, it seems that even this simple process has its complications. It has come to light that one of the biggest roadblocks in the study of quantum materials is the second step, where we try to predict a property of the material based on formulated Hamiltonian. Naturally, understanding why this is a difficult issue and what it implies physically is of great importance if we want to advance the field. It these kinds of questions that Quantum Hamiltonian Complexity attempts to grapple with.

Quantum Hamiltonian Complexity was born out of the insight that that a number of Hamiltonians of interest looked like variants of theoretical computer science problems. As a result, it was thought that understanding Hamiltonians using computer science tools, namely computational complexity theory, could bring new insights that traditional physics could not provide. This thinking ended up paying off, as the field
has made many breakthroughs in physics, mainly in characterizing things the amount and spread of entanglement in quantum materials. In this review of Quantum Hamiltonian Complexity, we will 1) set up the necessary background for understanding the field and 2) talk about field’s canonical problem, the $k$-local Hamiltonian problem. Along the way, we will highlight some important results and some implications for physics.

2 Quantum Hamiltonian Complexity

2.1 Classical Complexity Theory

Before we dive into Quantum Hamiltonian Complexity, we first need to understand the fundamentals of classical complexity theory and its quantum analogue.

Generally, computational complexity theory attempts rigorously understand the resources needed, typically in quantities of space (memory) and time (running time), for a model of computation to complete a desired problem. It may seem that the model of computation is vital to the study of the computational resources of any problem - what operations and resources are allowed changes what one could potentially do. However, counter-intuitively, the simple Turing Machine suffices as a model of computation that can not only simulate any deterministic program run on any digital machine, but can do so "efficiently" enough that its provably indistinguishable from any other model of computation (see the "Extended Church-Turing Thesis" for this \[5\]). The specific details of the Turing Machine are not too important in order to understand this paper’s span of complexity theory. However, it is helpful know the higher-level and other relevant definitions important to future discussion.

A Turing Machine (TM) takes in a $x = \{0, 1\}^*$ string from a function $f$ it wants to compute and follows an algorithm governed by a set of rules to output $f(x) = \{0, 1\}^*$. If the sum total number of rules used in the algorithm to compute $f(x)$ requires at most $T(|x|)$ for some $T : \mathbb{N} \rightarrow \mathbb{N}$, the given TM computes $f$ in $T(n)$ time. Typically, $T(n)$ are functions are of the form $n, n \times \log(n), 2^n, \text{etc.}$ and are written in asymptotic form (e.g. big-O notation). Since this paper will mainly deal with upper-bounds, the only asymptotic form we need to care about is big-O, which gives an upper bound on the running time of a computation. Formally, if $g(x) > 0$ for large $x$, $f(x) = O(g(x))$ if $|f(x)| \leq Mg(x)$ for some positive constant $M$ and at large values of $x$. Informally, $f(x) = O(g(x))$ means that $f(x)$ is upper-bounded by $g(x)$ as $x \rightarrow \infty$ \[4\].

Since, as we had stated earlier, TMs are postulated to efficiently simulate all deterministic models of computation, generally the smallest $T(n)$ that a Turing Machine needs to compute a function $f$ universalizes to state how much "computation" is necessary to perform the task. This is where complexity classes come into play. Complexity
classes are sets of problems that can be computed given the space and time resources characterized in that class. Since the inputs and outputs to problems are quite variable, any problem within a complexity class is talked about as a decision problem, where the original problem is formulated as binary question that requires computing the problem in order to answer (e.g. multiplication decision problem: is the kth bit of the product of inputs x and y 1?). The inputs with "yes" instances are the languages for those decision problems, and we state that a language L is in a complexity class if there exists an algorithm f the machine uses, where f : {0, 1}∗ → {0, 1}, such that f(x) = 1 ⇔ x ∈ L [4].

With complexity classes generally defined, we can state the most important complexity classes for our discussion here [4]:

**P** (Polynomial Time) - the class of languages (∈ {0, 1}∗) where there exists a deterministic Turing Machine that decides an L in the class in time α * T(n^c) for α, c ∈ R and α, c ≥ 1

**NP** (Non-deterministic Polynomial Time) - the class of languages where there exists a polynomial p : N → N and deterministic, polynomial time TM M (called a verifier for L) such that for every input x, we have

\[ x ∈ L ⇔ \exists u ∈ \{0, 1\}^{p(|x|)} \text{ s.t. } M(x, u) = 1 \]

Informally, the P complexity class contains all problems solvable in polynomial time (O(n^c) for c ≥ 1) while the NP complexity class contains all problems whose solutions can be checked in polynomial time by a TM given some polynomial-length proof (also called witness) u. It’s obvious that P ⊆ NP since a TM can verify a solution is correct in polynomial time by computing the solution in polynomial time. The infamous P = NP? is a longstanding problem in computer science, although wildly believed to not true.

Before moving on to quantum analogues of these complexity classes, it is important to discuss the concept of reductions, which will come up in later discussion. The formal definitions are not too important, but for a problem in class C, we say that that it is **C-Complete** if every other problem in the class C can be reduced to an instance of that problem in polynomial time. Similarly, a problem is considered **C-Hard** if a problem in the class of **C-Complete** problems can be reduced one of its instances in polynomial time. As a result, **C-Complete** are the hardest problems in class C and **C-Hard** problems are at least as hard as the hardest problems in class C. The essential distinction to make here about these two similar definitions is that **C-Complete** problems are a subclass in C while **C-Hard** problems don’t necessarily have to be in class C. Overall, both these types of problems are incredibly powerful, since if there exists a TM that solves a **C-Hard** or **C-Complete** problem in T(n), then all problems in C are solvable in T(n).
2.2 Quantum Complexity Theory

Although we stated before that Turing Machines are thought to be able to efficiently simulate all models of computation, it hasn’t been rigorously proven. In fact, the digital quantum computing model has put into question the validity of the Extended Church-Turing Thesis [5]. It is widely believed that there is no classical polynomial-time algorithm to find the factors of a number, but the famous Shor’s algorithm showed that it is possible in the quantum computational model. As a result, classical complexity classes clearly cannot extend to quantum computation - quantum analogues of classical complexity classes have to be defined.

The quantum analogue of \( P \) is \( BQP \), which stands for "Bounded-Error Quantum Polynomial Time". Since quantum computation is naturally probabilistic, the class \( BQP \) is defined similarly to \( P \), but with the caveat that it decides a given \( L \) for a problem in its class with error at most \( \frac{1}{3} \). Similarly, the quantum analogue for \( NP \) is the class \( QMA \) (Quantum Merlin-Arthur), which has a similar error bound for accepting or rejecting a proof for problems. In these complexity classes, the model of computation are either sets of quantum circuits with running time based on the number of sequential gates (depth) of the circuit or Quantum Turing Machines, which are Classic Turing Machines with additional rules. In either case, they encapsulate computability that extends to any quantum computation model.

Quantum complexity research, similar to its classical counterpart, deals with understanding what problems are in what quantum complexity classes. Another important task in the field is understanding how quantum complexity classes relate to their classical counterparts. The depiction on the next page offers one possible connection between the various complexity classes.

Quantum complexity theorists are not exactly sure what the boundary of \( BQP \) is. It is known that \( P \subseteq BQP \) and that \( BQP \subseteq \text{PSPACE} \) (the class of languages deicideable by a classical, deterministic Turing Machine with polynomial space and unlimited time). Both proofs are rather involved and include use and proofs about Quantum Turing Machines. Most notably, though, is that even though quantum computers are toted to be incredibly powerful, it is widely believed that they cannot solve \( \text{NP-Complete} \) problems in polynomial time ( alas, so close, yet so far!). \( BQP \)'s exact relation to \( \text{NP} \), however, is still not understood. For example, it's been shown by Aaronson that although quantum computers may not be able to solve \( \text{NP-Complete} \) problems in polynomial time, they can solve problems outside of \( \text{NP} \) itself (outside of the polynomial hierarchy) [2]. Finally, although not depicted in the diagram, \( QMA \) is known to be between \( \text{NP} \) and \( \text{PSPACE} \) (\( \text{NP} \subseteq QMA \subseteq \text{PSPACE} \)) [1]. More on quantum complexity classes and their relations, as well current research, can be read here [8].
2.3 Cornerstone of Quantum Hamiltonian Complexity - \( k \)-local Hamiltonian

2.3.1 Definition and Complexity Results

Now that we’ve learned the necessary details about classical and quantum complexity, we can move onto discussing how the topics relate to Hamiltonians.

While Hamiltonians can come in all shapes and sizes, in condensed matter systems where particles are generally restricted onto some geometrical structure, phenomenological Hamiltonians tend to be constructed from the particles’ local interactions with neighboring particles (the most notable examples of these are quantum spin systems \([6]\)). We call a Hamiltonian describing particles interacting with \( k \) other particles a \( k \)-local Hamiltonian. Formally, let a \( k \)-local Hamiltonian act on \( n \) qudits each with \( d \) discrete states \((|\Psi\rangle \in (\mathbb{C}^d)^\otimes n)\) on a Hilbert space \( \mathcal{H} \) of dimension \( d^n \). We have that for \( r \) local terms \( \{H_i\}_{i=0}^{r} \), \( H = \sum_{i=0}^{r} H_i = \sum_{i=0}^{r} (I_{[n] \setminus k_i} \otimes H_{k_i}) \), where each \( k_i \) is the unique \( k \)-length set of particle indices in \([n]\) that \( H_{k_i} \) acts on.

While this paper won’t do a deep dive into the explicit details of condensed matter physics, one of the most important tasks in studying a quantum system is to find its ground state. For many materials, their properties at room-temperature don’t diverge much from their properties at the ground-state energy (e.g. metals, who have their characteristics bound by Fermi-Dirac statistics don’t diverge much from these statistics until surpass their Fermi temperature of \( 10^4 T \)). Furthermore, many critical phenomena, like superconductivity, would like to best be understood at the materi-
als’ different regimes. Thus, understanding the complexity of calculating the ground state for a given Hamiltonian is incredibly important. Thus arose the infamous k-local Hamiltonian problem.

We give an explicit definition for the problem [6] below, formulated as a promise problem. Promise problems are generalizations of decision problems where instances are promised to be either YES or NO (promise problems are necessary when there are instances that are neither). The goal of computing any promise problem is to say whether a given instance is a YES instance or NO instance - similar to deciding the language for a decision problem.

**k-local Hamiltonian problem (k-LH)** - given as input a k-local Hamiltonian $H$ acting on $n$ qudits with $d$ different states with the constraint \{$H_i\}_{i=1}^r \subseteq \mathcal{H}(\mathbb{C}^d)^\otimes k$, and threshold parameters $a, b \in \mathbb{R}$, such that $0 \leq a < b$ and $(b - a) \geq \frac{1}{\text{poly}(n)}$, decide to output 1 or 0 based on the following, one of which is promised to be true about $H$’s eigenvalues \{$\lambda(H)$\}:

1. If there exists a $\lambda \in \{\lambda(H)\}$ such that $\lambda \leq a$, output 1
2. If all $\lambda \in \{\lambda(H)\}$ are such that $\lambda \geq b$, output 0

There are a lot of details in the above definition, but essentially, in order to differentiate between YES and NO instances of the k-LH problem, a machine has to compute the lowest eigenvalue of $H$, which is the ground-state energy for the system. Directly finding the eigenvalue decomposition for $H$ may seem like the appropriate algorithm, but since $H$ scales exponentially with the system size and the fastest eigenvalue decomposition algorithm runs in polynomial time [7], it is clearly not efficient. Fortunately, we can use complexity theory to better understand the resources needed to do this problem for a given $k$. Unfortunately, it is not good news.

We can easily show that the 3-local Hamiltonian over any $n$ qudits is NP-Hard and therefore intractable (inefficient to compute) for both classical and (likely) quantum computers. This can be done through the power of language reductions mentioned earlier, where we take a NP-Complete problem and convert it into an instance of the k-local Hamiltonian problem.

The most famous NP-Complete problem is the 3-SAT problem, which is a Boolean satisfiability problem defined as follows. Say we have $n$ terms denoted as $x_1, x_2, \ldots, x_n$, all which take binary values ($\{0, 1\}$). An instance of 3-SAT is of the form $\wedge_i^r (a \vee b \vee c)$, for $a, b, c \in \{x_1, x_2, \ldots, \neg x_1, \ldots, \neg x_n\}$. Each disjunction between three variables is called a clause and each variable only appears in a single clause. To solve an instance of the problem is to find a satisfying assignment of $\{0, 1\}$ to each of the variables (including the negations) so that all the clauses are true. This seemingly simple problem is intractable and is generally the goto for problem reductions.

We can construct a k-LH instance from any given 3-SAT instance in the following way.
For any given 3-SAT instance, let a given clause be given by \(c_i\). We can construct a 3-local Hamiltonian \(H\) by having each local term \(H_i\) be

\[
H_i = \sum_{x \in \{0, 1\}^3, \text{s.t. } c_i=0} |x\rangle \langle x|
\]

Let our Hilbert space be of size \(2^n\). We have that our eigenstates (\(\{|\psi\rangle\}\)) of \(H\) are the \(2^n\) orthonormal binary string kets that span the Hilbert space. Effectively, we have that if a \(c_i = 0\) for the assignment associated with \(|\psi\rangle\), then \(\text{Tr}(H_i |\psi\rangle \langle \psi|) > 0\). If all clauses are satisfied \((c_i = 1)\) for the binary assignment \(|\psi\rangle\), then \(\text{Tr}(H_i |\psi\rangle \langle \psi|) = 0\). Since \(|\psi\rangle\) such that \(\text{Tr}(H |\psi\rangle \langle \psi|) = 0\) means that it is the ground state (or part of the ground space), finding it is equivalent to finding a (or multiple) string assignments such that all \(c_i = 1\). Thus, since any 3-SAT problem can be formulated as a specific case of 3-LH problem, it must be that 3-LH is as hard as 3-SAT to compute and therefore \(\text{NP-Hard}\).

It should be noted for this particularly instance, the eigenstates of the Hamiltonian were product states and since they were the eigenstates for all the local terms as well, all the \(\{H_i\}\) could be diagonalized simultaneously in the given basis. Generally, a \(k\)-LH problem instance where the the \(\{H_j\}\) are simultaneously diagonalizable in the given basis or the eigenstates are product states are equivalent to classical Boolean satisfiability problems. The real (quantum) difficulty of the \(k\)-LH problem comes from the fact that states of the Hamiltonian can be entangled.

So we can quickly say that 3-KH problem is pretty hard, but can we say anything more generally. We can! We can make things more rigorous and prove that the \(k\)-LH problem for \(k \geq 2\) is \(\text{QMA-Complete}\) (1-LH problem is in \(\text{P}\), simply because one can optimize over each \(H_i \in H\) independently). The proof is rather involved and contains some computer science concepts outside the scope of this paper, but we can at least get part of the way through the proof by showing that the \(k\)-LH problem is \(\text{QMA}\). To do this, we need to build a quantum polynomial verifier circuit that is able to, given an instance \((H, a, b)\), determine with high accuracy on a given quantum proof state whether the instance is a \(YES\) or \(NO\) instance.

Suppose we have an instance \((H, a, b)\) where \(H = \sum_{j=1}^{r} H_j \in \mathcal{H}((\mathbb{C}^2)^{\otimes n})\). Suppose we are given the ground state \(|\nu\rangle\) as proof that the \((H, a, b)\) instance is a \(YES\) instance. Let us first transform the given proof into the following state \(|p\rangle \in (\mathbb{C}^2)^{\otimes n} \otimes \mathbb{C}^2\)

\[
|p\rangle = |\nu\rangle \otimes |0\rangle
\]

Let us call first register of \(|p\rangle\) (\(|\nu\rangle\)) the proof register. The second register is the answer register. When we submit this proof to our constructed verifier circuit, we will measure the answer register and state that we’ve received a \(YES\) instance if we’ve measured \(|1\rangle\) and \(NO\) instance if we’ve measured \(|0\rangle\).
Now let us construct our verifier circuit $V$. Ideally, we would like to have our circuit act on $|p\rangle$ so that, in the answer register, the state $|1\rangle$ has a high probability amplitude in the YES case and a low probability amplitude in the NO case. To do this, we can create a verifier operator that correlates the answer register to the actual eigenvalues of the Hamiltonian. Let’s see how this is done.

Now, since $H$ is Hermitian, we can write each $H_j$ in its spectral decomposition. Let a given $H_j$ be $\sum_s \lambda_s |\psi_s\rangle \langle \psi_s|$. Note that since $H_j$ is Hermitian, its eigenvectors span the space of $\mathcal{H}(\mathbb{C}^2)^{\otimes n}$. We can create an associated operator $V_j : \mathcal{H}(\mathbb{C}^2)^{\otimes n} \otimes |0\rangle \mapsto \mathcal{H}(\mathbb{C}^2)^{\otimes n} \otimes \mathcal{H}(\mathbb{C}^2)$ with the following action

$$V_j(|\lambda_s\rangle \otimes |0\rangle) = |\lambda_s\rangle \otimes (\sqrt{\lambda_s} |0\rangle + \sqrt{1-\lambda_s} |1\rangle)$$

We can write $|\nu\rangle$ in the eigenbasis of $H_j$ (the constants don’t matter here). With $V_j$ acting on $|p\rangle$, we can see that the probability of measuring $|1\rangle$ is $1 - \langle \nu | H_j | \nu \rangle$.

$$V_j(\sum_s \alpha_s |\lambda_s\rangle \otimes |0\rangle) = \sum_s \alpha_s (|\lambda_s\rangle \otimes (\sqrt{\lambda_s} |0\rangle + \sqrt{1-\lambda_s} |1\rangle))$$

Let the RHS be $|\eta\rangle$ for simplicity sake. We have

$$Pr(1) = |(I \otimes |1\rangle \langle 1|) |\eta\rangle|^2 = \langle \eta | (I \otimes |1\rangle \langle 1|) |\eta\rangle$$

Again, since $H_j$ is Hermitian, we have that its eigenbasis is orthonormal. As a result, we have

$$\langle \eta | (I \otimes |1\rangle \langle 1|) |\eta\rangle = \sum_s \alpha_s^* \alpha_s (1 - \lambda_s)$$

$$\sum_s \alpha_s^* \alpha_s (1 - \lambda_s) = 1 - \langle \nu | H_j | \nu \rangle$$

Thus, we have what we want. However, we truly care about $\langle \nu | H | \nu \rangle$. Since $\sum_j \langle \nu | H_j | \nu \rangle = \langle \nu | H | \nu \rangle$, we can do the following to get the probability of measuring $|1\rangle$ in the form of $1 - \langle \nu | H | \nu \rangle$.

Let us attach an index register to $|p\rangle$ in the form of $(\frac{1}{\sqrt{r}} \sum_{j=1}^r |j\rangle \langle j|)$, where $\{j\}$ is an orthonormal set corresponding to indices $(1, 2, 3..., r)$. Now we have that $|p\rangle$ is

$$|p\rangle = \frac{1}{\sqrt{r}} \sum_{j=1}^r |j\rangle \langle j| \otimes |\nu\rangle \otimes |0\rangle$$

Now, let us our define our verifier circuit $V$ as the following

$$V = \sum_j |j\rangle \langle j| \otimes V_j$$

It is easy to see that $V$ acting on $|p\rangle$ now yields $\frac{r - \langle \nu | H | \nu \rangle}{r} = 1 - \frac{\langle \nu | H | \nu \rangle}{r}$. Now, with this verifier circuit, the probability of stating YES when an instance is YES is lower
bounded is $1 - \frac{a}{r}$ and the probability of stating $\text{YES}$ when an instance is $\text{NO}$ upper bounded by $1 - \frac{b}{r}$. By repeating the verifier circuit on multiple copies of the proof (where we still maintain a polynomial length circuit and proof), we can bring $1 - \frac{a}{r}$ above $\frac{2}{3}$ and $1 - \frac{b}{r}$ below $\frac{1}{3}$ as desired [6]. With this, we have effectively shown that the $k$-LH problem is in $\text{QMA}$.

We have captured the complexity of the $k$-LH problem, but there are a couple of more things worth mentioning about the complexity of the estimating the ground state of a Hamiltonian. Truth be told, finding the ground state of a Hamiltonian is actually even harder than what we’ve worked with here. In the definition given for the $k$-LH problem, we’ve stated there is at least an inverse polynomial gap between the thresholds $b$ and $a$ ($b - a \geq \frac{1}{\text{poly}(n)}$). If $b - a$ were allowed to be closer together, then it would difficult to discern $\text{YES}$ and $\text{NO}$ instances. Even worse, the $k$-LH problem is only $\text{QMA-Complete}$ if the spectral gap (the difference between the smallest two eigenvalues) is inversely polynomial [6]. If the ground state and first excited state of a Hamiltonian happen to exponentially close in energy, then determining the ground state of a Hamiltonian would be much harder than a $\text{QMA-Complete}$ problem.

All this is quite disheartening, but one may be interested to what extent a relaxation of the gap between $a$ and $b$ in the $k$-LH problem (which effectively connects to the precision of the approximation of the ground state) means for the system’s computational complexity. This is where the the Quantum PCP Conjecture comes in.

### 2.3.2 Quantum PCP Conjecture

The Quantum PCP conjecture is an analogue of the classical version, which states that for 3-SAT, it is still $\text{NP-hard}$ to satisfy a ratio of the clauses proportional to the total number of clauses. Informally, the Quantum PCP conjecture posits that approximating the ground state of a local Hamiltonian with error proportional to the size of the system (the number of local terms in the Hamiltonian) is just as hard as the original $k$-LH problem. Formally, in relation to the $k$-LH problem, the Quantum PCP conjecture is [3]:

**Quantum PCP Conjecture (by gap amplification)** - Given a $k$-LH problem with $m$ local terms and gap $b - a = \gamma * m$ for $0 < \gamma < 1$, we have that it is $\text{QMA-hard}$ (under polynomial time reductions).

The idea behind this is similar to the reduction of 3-SAT we used to show that the $k$-LH problem is $\text{NP-hard}$. Say that there exists a quantum polynomial transformation (circuit) that takes an instance ($\sum_i^m H_i$ with $(b - a) = \delta = \frac{1}{\text{poly}}$) and transforms it into another instance with $\sum_i^m H'_i$ and $\delta = \gamma * m$ for $0 < \gamma < 1$. If the latter problem was of lower complexity than $\text{QMA-hard}$ and could be solved, then we
could reduce any $k$-LH problem to it and effectively solve the original $k$-LH problem. Since we believe that the original $k$-LH problem is \textbf{QMA-hard} however, it must be that the relaxed version of the problem is \textbf{QMA-hard} as well.

This conjecture if proven true, would again be quite unfortunate for physicists and computer scientists around the world. However, true or not, the conjecture has amazing implications for quantum systems, namely involving entanglement. It is generally thought that entanglement in systems only becomes noticeable at extremely low temperatures $T \approx 0$. As a result, it is expected that systems that equilibrate at $T > 0$ in the Boltzmann distribution are thought to have no entanglement. The advent of the Quantum PCP conjecture being generally true, however, could mean that, given some fine-tuned parameters, there are no product states at a energy threshold above $E_0$ (i.e. states are still highly entangled at a certain point above $T > 0$). Another interesting, stronger point made in a survey about this conjecture is that the Quantum PCP conjecture being true would mean that entanglement \textit{must} play a crucial role in a system reaching equilibrium at room temperature. For these reasons (and many more \cite{3}), the Quantum PCP conjecture is at the forefront of Quantum Hamiltonian Complexity research.

2.4 Developments of Interest

Quantum Hamiltonian Complexity is an ever-growing field. A lot of research has been devoted to various relaxations to the $k$-LH problem that more resemble actual Hamiltonians of different materials. For example, an important model for understanding topological materials is the toric code, where electrons on a square 2D lattice locally interact with neighboring electrons. Due to its construction, the local terms of the Hamiltonian pairwise commute with each other. It was found that allowing the local terms to commute with each other in the $k$-LH problem made finding the ground state only in \textbf{NP} \cite{6}.

Another interesting notion of capturing the complexity of quantum systems is through the use of \textit{area laws}, which state that entanglement between a subset of particles of a system’s ground state scales proportionally with the particles at the boundary of the partition and not the number of particles in the subset itself. Finding area laws for certain systems has both greatly reduced the complexity in estimating the ground state and also characterized the nature of entanglement in those systems. Of utmost interest now is to find area laws for 2-dimensional systems (e.g. particles on a square lattice) \cite{6}.

Overall, if you’re interested in Quantum Hamiltonian Complexity, please refer to the references used in this review. What we’ve covered is small sliver and there is much more to explore!
References


